Steerable AI-powered Art-making Tools

by

John Joon Young Chung

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2023

John Joon Young Chung

jjyc@umich.edu

ORCID iD: 0000-0002-8492-2525

ACKNOWLEDGMENTS

I would like to thank my advisor, Eytan Adar, who supported my research vision. With his advice, I could learn the joys of doing research, and pursue my vision of supporting art-making with technologies. Eytan was also supportive during difficult times, and I could not reach this point of the research journey without his help. I also want to thank my dissertation committee members, prof. Nikola Banovic, prof. Sophia Bruekner, Dr. Mira Dontcheva, and prof. Anhong Guo, for their support on this work. I am also grateful to all the others who helped me throughout this journey: prof. Juho Kim, Dr. Minsuk Chang, prof. Eric Gilbert, prof. Sungsoo (Ray) Hong, prof. Sangwon Lee, Dr. Rubaiat Habib, Dr. Saleema Amershi, Dr. Ece Kamar, prof. Jean Young Song, prof. Yan Chen, Dr. Sai R. Gouravajhala, prof. Licia He, prof. Sindhu Kutty, Dr. Hijung Valentina Shin, prof. Haijun Xia, Dr. Li-yi Wei, Dr. Kang Min Yoo, Dr. Hwaran Lee, prof. Kenneth Huang, prof. Dongyeop Kang, Dr. Katy Gero, and prof. Jung Yup Lee. I also want to thank my colleagues and collaborators, Jordan Huffaker, Rebecca Krosnick, Zhefan Ye, Anthony Liu, Yiwei Yang, Ingab Kang, Youngwook Do, Hee-Seung Moon, Divya Ramesh, Jaylin Herskovitz, Harman Kaur, April Wang, Chuan Yan, Kihoon Son, Wooseok Kim, Taesoo Kim, Hyungyu Shin, Hyunwoo Kim, Taewook Kim, Yoonjoo Lee, Charvi Rastogi, Gregory Croisdale, Nel Escher, Snehal Prabhudesai, Anindya Das Antar, Sumit Asthana, Jane Im, Wooseok Seo, Xin Qian, Mina Lee, and many others. I should also thank Calmdown-man and his friends, who made me withstand a long and lonely pandemic with video content that always made me laugh. My family and friends also have been always supportive of my Ph.D. journey. I am more than grateful to my mother, Jinah Kang, and my father, Jintai Chung, my sister, Jiyoung Chung, my brother-in-law, Sangwoo Ahn, my niece, Seeun An, and our cute doggo, Choco. Last but not least, I want to thank my partner, Jeongyeon Kim, who encouraged me during hardships and became one reason for me to go through the journey beyond my Ph.D.

TABLE OF CONTENTS

ACKNOWLEDGMENTS . . . ii

LIST OF FIGURES . . . vii

LIST OF TABLES . . . xiii

ABSTRACT . . . xiv

CHAPTER

1 Introduction . . . 1

1.1 AI-powered Art-making Tools . . . 2
1.1.1 Creativity Support Tools and AI-powered CSTs . . . 2
1.1.2 Human-AI Systems . . . 3

2 Artist Support Networks: Implications for Future Creativity Support Tools . . . 5

2.1 Introduction . . . 5
2.2 Background and Related Work . . . 7
2.2.1 Human Support for Art Creation . . . 7
2.3 Research Method . . . 8
2.3.1 Participants . . . 10
2.3.2 Interview Protocols and Data Generation . . . 10
2.3.3 Analysis Method . . . 11
2.4 Results . . . 11
2.4.1 Support Provided in the Artist's Support Network . . . 12
2.4.2 Relationships and Dynamics . . . 14
2.4.3 Frictions and Conditions for Success . . . 20
2.5 Discussion . . . 23
2.5.1 Informing the Design of CSTs . . . 23
2.5.2 Comparison to Previous Models on Artistic Support . . . 28
2.5.3 Limitations and Future Work . . . 30
2.6 Conclusions . . . 30

3 The Intersection of Users, Roles, Interactions, and Technologies in Creativity Support Tools . . . 32

3.1 Introduction . . . 32

3.2 Background and Related Work . . . 33
3.3 Literature Review Method . . . 34
3.3.1 Sampling . . . 35
3.3.2 Coding Process . . . 36
3.4 Codes . . . 37
3.4.1 Roles of CSTs . . . 37
3.4.2 Interaction Approaches Used by CSTs . . . 39
3.4.3 Technologies for CSTs . . . 40
3.4.4 Users of CSTs . . . 42
3.5 Coding and Analysis Results . . . 43
3.5.1 Distribution for Each Taxonomy . . . 44
3.5.2 Within-Taxonomy Analysis . . . 44
3.5.3 Cross-Taxonomy Analysis . . . 46
3.6 Discussion . . . 51
3.6.1 Design Patterns of AI-CSTs and Their Limitations . . . 52
3.6.2 Trends in CST Research for Art-making . . . 53
3.6.3 How Roles Are Supported with Interaction Approaches and Technologies 54
3.6.4 Patterns in Supporting Users . . . 55
3.6.5 Comparison to Previous Work . . . 55
3.6.6 Limitations and Future Work . . . 56
3.7 Conclusion . . . 57

4 Steerability in AI-powered Art-making Tools . . . 58

4.1 Definition and Need for Steerability . . . 58
4.2 Requirements of Steerability for AI-CSTs . . . 58
4.2.1 Facilitating Iterative Steering . . . 59
4.2.2 Efficient Steering . . . 59
4.3 Design Approach: Mixing Familiar Input Modalities . . . 60

5 *Artinter*: Steering Search and Generation with Mood Board and Slider Interactions in Visual Art Commission . . . 62

5.1 Introduction . . . 62
5.2 Background and Related Work . . . 65
5.2.1 Artistic Communication Means as Boundary Objects . . . 65
5.2.2 Tools for Searching Artifact Instances . . . 66
5.2.3 Tools for Artifact Generation . . . 67
5.3 The Art Commission Process . . . 67
5.3.1 Findings . . . 68
5.3.2 Design Goals . . . 69
5.4 Commission Communication with *Artinter* . . . 70
5.4.1 Set-up . . . 71
5.4.2 Concept Building (D1) . . . 72
5.4.3 Artifact Expansion . . . 74
5.4.4 Implementation . . . 78
5.5 Study 1: Support of Design Goals . . . 78

5.5.1 Participants 78
5.5.2 Procedure 78
5.5.3 Results 79
5.6 Study 2: Usage Patterns 84
5.6.1 Participants 85
5.6.2 Procedure 85
5.6.3 Results 86
5.7 Discussion 88
5.7.1 Mechanisms to Ground Concepts 88
5.7.2 Iteration for Artistic Communication 90
5.7.3 Tension in Generative AI 90
5.7.4 AI-CSTs in Unbalanced Collaborative Contexts 91
5.8 Conclusion 92

6 *TaleBrush*: Steering of Story Generation with Visual Sketching 93

6.1 Introduction 93
6.2 Related Work 96
6.2.1 Writing Support Tools 96
6.2.2 Machine Story Generation 97
6.2.3 Visualizing and Visually Expressing Stories 98
6.2.4 Sketching Interactions 98
6.3 Sequence Steering Through Sketching 99
6.4 *TaleBrush*: Interface 101
6.4.1 Iterative Co-creation With Line Sketching 101
6.4.2 Interface 102
6.4.3 Implementation 103
6.5 *TaleBrush*: Technical Details 104
6.5.1 Collecting Fortune of the Protagonist 104
6.5.2 Technical Architecture 106
6.6 Technical Evaluation 112
6.6.1 Fortune Recognition Evaluation 112
6.6.2 Automated Generation Evaluation 113
6.6.3 Human Evaluation on Generation 114
6.7 User Study 116
6.7.1 Participants 116
6.7.2 Procedure 116
6.7.3 Results 117
6.8 Discussion 120
6.8.1 Sketching to Address The Gulf of Iterative Human-AI Co-Creation . . . 120
6.8.2 Generalizing Line Sketching-based Steering 120
6.8.3 Disagreement Between Human and AI 121
6.8.4 Further Support in Story Co-creation 121
6.8.5 Limitation and Future Work 122
6.9 Conclusion 123

7 *PromptPaint*: Steering of Text-to-Image Generation Through Paint Medium-like Interactions . . . 124

7.1 Introduction . . . 124
7.2 Background and Related Work . . . 126
7.2.1 Painting-From Physical to Digital . . . 126
7.2.2 AI Image Generation . . . 127
7.2.3 Interaction with AI Generation . . . 128
7.3 Interacting with Generative Models Like Paint Medium . . . 129
7.3.1 Mixing Colors: Exploring Vector Spaces . . . 129
7.3.2 Applying Colors onto Canvas: Gradual Generation . . . 131
7.4 *PromptPaint*: Interface . . . 133
7.4.1 Canvas and Basic Editing Functions . . . 133
7.4.2 T2I Generation Functions . . . 133
7.5 *PromptPaint*: Technical Details . . . 138
7.5.1 Background: Diffusion-based Text-to-Image Generation Models . . . 138
7.5.2 Prompt Vector Manipulations . . . 139
7.5.3 Latent Manipulation . . . 139
7.5.4 Implementation . . . 140
7.6 User Study . . . 141
7.6.1 Participants . . . 141
7.6.2 Procedure . . . 141
7.6.3 Result . . . 142
7.7 Discussion . . . 148
7.7.1 Generalizing Paint-Medium-Like Interactions for Generative Tools . . . 148
7.7.2 In-generation Interaction for T2I models . . . 149
7.7.3 Design Trade-Offs in Generative Tools . . . 150
7.7.4 Ownership of Generated Artifacts . . . 150
7.8 Conclusion . . . 151

8 Reflections . . . 152

8.1 Designing Steering Interactions for AI-powered Art-making Tools . . . 152
8.1.1 Reflection: Applying Design Approaches to Other Domains . . . 153
8.2 Future Directions . . . 154
8.2.1 Steering for More Complex Mediums . . . 154
8.2.2 Leveraging Artificial Intelligence Further in AI-CST . . . 154
8.2.3 Steerable AI for New Types of Artifacts . . . 155
8.3 Conclusion . . . 155

APPENDICES . . . 157

BIBLIOGRAPHY . . . 162

LIST OF FIGURES

FIGURE

2.1 A summary of the support types identified through our study. Artifact-influencing support directly contributes to artifacts, while artist-influencing support impacts the artist or the creation process. The arrows on the left indicate possible feedback and controls. Though the dynamics vary, artifact-influencing support (i.e., implementation and ideation) are influenced by the artist (self), their collaborators (actors), or tools around artists and the actors. Among the artist-influencing support, managing impacts creation through the management of actors and tools (a meta-role), while teaching impacts by artist's self-improvement. 12

2.2 I describe socio-artifact networks with relational graphs of PEOPLE and ARTIFACTS. Artists and actors provide the support that impacts artifacts (ideas and implementation, with blue edges) or each other (ideas and teaching, with red edges). Actors also can provide 'managing' support to the artist and other actors (purple edges). The amount of an idea or implementation each person provided to the artifact is color-coded in the artifact diagram. The contributions made by the artist are in light teal for ideas and dark teal for implementation. For contributions from the actor, I use light yellow for ideas and dark yellow for implementation. The gradient indicates a 'range' of values. 15

2.3 Diagram of relationships in support network. Support relationship types are described as relational graphs between artists, actors, and artifacts. The width of edges indicates the amount of provided support. The amount of idea- and implementation-wise contributions made by the artist and the actor to the artifact is denoted with gradient glyphs. 16

2.4 Comparison of taxonomy on types of support. 28

2.5 Comparison of taxonomy of approaches for successful support relationship. 29

3.1 Co-occurrences of codes within each taxonomy. The number in each box indicates the ratio of tools that have the element in the column among all tools with the element in the row. On the y-axis, numbers in parenthesis indicate the number of CSTs with the code. In a), 'Idea' and 'Execution' stand for idea generation and execution assistance, respectively. In b), 'Pred' and 'Unpred' stand for predictable and unpredictable, respectively. 44

3.2 Percentage of interaction approaches (directness, predictability, and output) and technologies according to resource roles (items on the y-axis). On the y-axis, numbers in parenthesis indicate the number of CSTs with the code. 46

3.3 Percentage of interaction approaches (directness, predictability, and output) and technologies according to process roles (items on the y-axis). On the y-axis, the number of CSTs with the code is in the parenthesis. On the y axis, “Idea” stands *idea generation* and “Execution” is for *execution assistance*. 47
3.4 Percentage of roles according to user groups (items on the y-axis). On the y-axis, the number of CSTs with the code is in the parenthesis. In b), “Idea” stands for idea generation and “Execution” stands for execution assistance. 50
3.5 How each user group (on the y-axis) is supported with different interaction approaches (directness, predictability, and output) and technologies. On the y-axis, numbers in parenthesis indicate the number of CSTs with the code. 51

4.1 Steering interactions can bring AI results closer to the user’s expectations. 59
4.2 Dynamics of artist using AI tools 60

5.1 Design of *Artinter*. *Artinter* is a collaborative AI-powered system that supports communication around art commissions. It allows artists and clients to have shared boundary objects (1) on the mood board. With shared objects, *Artinter* allows users to have agreement on what they mean through concept building (2), or defining concepts with artifact instances. The information elicited from the development of concepts is also used to train *Artinter* to learn these user-defined concepts (3). Based on the learned concepts and shared artifacts, *Artinter* provides two AI-powered supports in expanding artifact instances (4): search and generation. 63
5.2 *Artinter* interface. *Artinter* is composed of a sketch pad (a) and a mood board (b). On the sketch pad, users can do raster image editing, such as brushing, erasing, or lassoing (a-1). Users also can copy-paste images on the mood board onto the sketch pad. With layers, users can add, remove, reorder, and hide them (a-3). On the mood board, users can add art images, texts, and color swatches (b-1). Note that color swatches are considered similar to art images except that they have a single color and the color is selectable. With art images or color swatches, users can define concepts (b-2). On the sketch pad, the user can also combine styles of images on the mood board and apply them to one of the layers (a-2, 4, 5, and 6). The example result of the style mix is presented in a-5. On the mood board, users can collaboratively search or generate artifact instances with concepts defined by themselves (b-3). They can also know who is online on the boards (c). The user also can expand a certain panel if they want to focus on working on one (d). The current image on the sketch pad can be moved to the mood board (e). 71
5.3 Building and updating concepts in *Artinter*. a) A user can create a concept by clicking the CREATE A CONCEPT button and inputting a textual name for the concept. b) While a user has selected a concept, the user can relate the concept to other ones by clicking R buttons showing up on other concepts (left). The user can also ‘unrelate’ already related concepts by clicking U buttons on concepts connected to the selected concept (right). c) The user can add a selected image (bottom-right) to a concept that does not have the image by clicking + button (left). It also can be removed from a concept that already includes the image by clicking - button (right). 72
5.4 Pipeline for acquiring vector representation for recognition and search function. 73
5.5 Iterative process of updating concepts in *Artinter*. 74

5.6 Concept controls for search and generation function. 75
5.7 Impact of scale and weight parameters in sketch generation function on a sketch pad. . 76
5.8 Study 1 survey results. 80
5.9 Participants defined concepts that can explain the artists' styles or clients' preferences. Participants could also collaboratively search for references with envisioned style-wise characteristics by adjusting sliders for user-defined concepts ("harsh lines", "chair", and "woods"). Artwork under "harsh lines" are by Jesse Hughes (A2), and those under "chair" and "woods" are provided by the C2. 81
5.10 Participants iterated on the generation by controlling available parameters and reached the desired one. Participants built upon intermediate or final generation results by pointing out preferred characteristics or adding sketches (as in the red circle). Artwork under concepts are by Danielle Moses (A1), and C1 provided the query image. 82
5.11 The case of the user's mental model not aligning with the AI's result. A5 expected the generation function to apply uniformly flat coloring as the style image, but the result showed a bit of a mix of colors in the background and other parts of the image. The image on the Content is by Rory Lucey (A5). 84
5.12 Creativity Support Index results. *, **, and *** indicate significant difference with $p < 0.05$, $p < 0.01$, $p < 0.005$, respectively. The error bar indicates the standard deviation. 86

6.1 *TaleBrush* uses a line sketching interaction for intuitive control and sensemaking of story generation with GPT-based language models, closing the gap in the iterative co-creation process between humans and AI. In our prototype, the protagonist's 'fortune' is controllable. The writer first decides the protagonist's name (A1), writes a portion of a story (A2), and then sketches how the protagonist's fortune should change in the story (A3, green shaded area). The x position of the sketched line indicates the chronological position in the story (sentences are the units). The y position shows how good or bad the protagonist's fortune should be (higher, better). The width of the line indicates the possible variance in the fortune of the generated sentences. Given the line sketch, *TaleBrush* will generate story sentences (B1, indicated with blue) and visualize the result on the original sketch (B2, the blue line and dots). The writer can always directly edit the generated text. They can also iterate by generating new text for the same sketch (B3, clicking on 'Generating Again') or revising their sketch and generating new text. Additional options allow the writer to specify how 'surprising' the generation should be (B4) or using the eraser tool (B5) to erase their sketch. Where the line is erased, *TaleBrush* generates unconstrained sentences. A history dropdown (B6) allows the writer to browse previously generated sentences. 94
6.2 Kurt Vonnegut's sketched line drawing of *Cinderella's* fortune over the course of her story [408]. 95
6.3 More than one story attribute can be expressed with a) multiple steering panels or b) the use of sketching speed. 100

6.4 *TaleBrush* allows writers to specify the level of 'surprise' for each generated sentence. A separate canvas can be opened where the writer can sketch a thin green line indicating the desired level of surprise. Here, we see a fortune arc for Mike (top) starting high, going low, and ending high. A corresponding surprise sketch (bottom) indicates that the author wanted the first part of the story to be more controlled but the end should be very surprising. 104
6.5 *TaleBrush* stores generated stories in a list (A) and allows writers to compare them on the canvas by visualizing the fortune arcs of all stored stories in low opacity (B). The writer can hover over stored stories in the list to see their text and story arc visualization. 105
6.6 Interface for annotating the protagonist's fortune level. The annotator can make comparisons between sentences in the given story. Calibration sentences help for making comparisons between stories. 106
6.7 The technical architecture of *TaleBrush*. To serve recognition and generation, the prompts of a Base-LM are tuned. These prompts are used interchangeably for a single LM. For fortune recognition, context, prompt embeddings, and target sentence are given as inputs and the fortune level is predicted as a real value between 0 and 1. For steerable generation with the fortune sketch, a modified version of GeDi is used. GeDi is a fine-tuned model that steers the generation according to real-valued fortune input, or continuous control code. Context and prompt embeddings are also used as input. For a sentence to be generated, the y-position of a point in a sketch is used as a continuous control code. This continuous control code has a value between 0 and 1 and its input embedding is interpolated with three class codes (bad for 0, neutral for 0.5, and good for 1) before being input into the modified GeDi. Base-LM used GPT-Neo while the modified GeDi used GPT2-medium. 107
6.8 How the protagonist's fortune is recognized in the summary of *Cinderella*. The visualized recognition results are similar to Vonnegut's original (Figure 6.2). 108
6.9 *TaleBrush* performs three story generation tasks: a) continuation, b) infill from previous, and c) infill from next. For different generation tasks, how contexts and prompts are structured for inputs differs. In *Input to GPT-Neo*, dark boxes indicate continuous prompts. In *Generated*, red text stand for generated sentences. 109
6.10 An approach to turn continuous control codes into embeddings that can be input to CC-LM. For three extreme classes (1, 0.5, 0), we use embeddings of the tokens, *good*, *neutral*, and *bad*, respectively. For other control codes, I interpolated the embeddings of these extreme classes. 111
6.11 Automated evaluation results on steered story a) continuation and b) infilling. Median of perplexity and steering difference (between given steering input value and output fortune) are plotted for the range between 25^{th} and 75^{th} percentile. In the regeneration condition, we measured the rate of sentences that were regenerated to minimize steering difference. The x-axis is presented in log-scale. 113

6.12 Human technical evaluation results on a) continuing and b) infilling generation. For steerability, the accuracy (A) in estimating which sentence is generated with a higher steering input value is reported. For coherence, grammaticality, and novelty, the ratio of evaluators choosing which condition has better quality is reported (CX for generation without steering, S for quality being similar, and CO for generation with steering). 115
6.13 P2's work: a) Edit of a generated story. Bolded, red, and blue text indicates initial context input, removed parts, and added texts, respectively. b) Line sketch and fortune arc of the generated story before editing. c) Fortune arc of the edited story. 117

7.1 *PromptPaint* allows flexible steering of diffusion-based text-to-image generation by combining prompt-based generation with paint medium-like interactions (e.g., oil painting, watercolor). After the user defines their prompts of interest (b), the tool allows users to mix prompts like how we mix colors of paint mediums by 1) interpolating them on the prompt palette (c, *prompt mixing*) or 2) adding extrapolating attributes (d, *directional prompt*). Moreover, *PromptPaint* allows gradual shaping of the artifact by allowing 1) changing of prompts during generation (c-1, g, *prompt switching*) and 2) spatial selection of area to generate image content (f, *prompt stencil*). *PromptPaint* also allows detailed specification of generation with configuration widgets (e) along with other image editing functions like moving image content, brushing, erasing, lassoing, and layer edits (a, h). 125
7.2 Mapping interactions of exploring vector spaces to interactions of mixing paint colors. Each of the discrete semantics (which would be categories, prompts, or examples and are represented as black squares in Vector) can map to discrete colors in paints (tubes in Paint). Then, the user can explore vector semantics between discrete semantics (black dots in Vector) in a way similar to how they would explore colors by mixing (brushes in Paint). 130
7.3 Using palette interaction for semantic mix has benefit in that the interpolation and the extrapolation can be represented in the same interface. 131
7.4 Mapping interactions of generation modularization to interactions of gradually painting an artifact. Within-generation interventions would correspond to multiple strokes applied on the same canvas area, and generating the artifact part-by-part would map to drawing the image part-by-part. 132
7.5 Interactions to mix and detach prompts in Prompt Palette. 134
7.6 Example results of prompt mixing. 134
7.7 Example results of directional prompts. The center result is the result without a directional prompt and the left and the right are results of applying directional prompts. The rightmost and leftmost results applied the full vector difference between the two end prompts. 135
7.8 With a prompt stencil, the user can specify the area of generation (dark grey). When the user completes brushing the area, the tool starts generating a part of the image while showing the process to the user. 135

7.9 When applying prompt stencil upon the existing image, *PromptPaint* considers those existing images for generation. For the area where the stencil is overlapping with the existing image, *PromptPaint* generates a new image that is similar to the existing image based on the overcoat value (higher, less similar). 136
7.10 Example results of overcoating generation. In this figure, the image is generated again from the far left image with varying overcoat values (more right, higher overcoat values). Images generated with higher overcoat values tend to be less similar to the original image. 136
7.11 Example results of prompt switching. Except for the first column which did not change the prompt during the generation (i.e., used the same prompt throughout the whole generation), each column switched the guiding prompt at a different point of the generation process. 137
7.12 The pipeline of diffusion-based text-to-image models. The technical manipulation of *PromptPaint* happens in the green dot, either by manipulating vector-embedded prompts or intermediate latent. 138
7.13 The histogram of how participants answered the creativity support index questions. The high score indicates that the participant perceived that *PromptPaint* supports the criteria. 142
7.14 The histogram of how participants answered the question about the sense of ownership, contribution, and collaboration. The higher the scores, the participant felt that they have more ownership than *PromptPaint*, they contributed more than AI, and they collaborated with AI functions. 143
7.15 Comparison of four different functions on if they helped with 1) controlling or 2) exploring generation. `mix`, `dir`, `swit`, and `sten` stand for *prompt mixing*, *directional prompt*, *prompt switching*, and *stencil*, respectively. 143
7.16 Prompt mixing from P5. By mixing semantically close prompts, the user can control the generation to explore the image space in between. 144
7.17 Prompt mixing from P1. For semantically far prompts, the mixing prompts linearly in the vector space sometimes did not result in a linear impact on the generated images. . 144
7.18 Directional prompt from P6. 145
7.19 Prompt switching from P2. 145
7.20 Prompt stencil from P2. 146

A.1 Accuracy of recognizing concepts with the varying number of artifact instances and trained concepts. Shaded areas indicate the range of one standard deviation in accuracy. 158
A.2 Two cases of how relating relevant concepts impact the performance of recognition and search, for *blue-red* and *rough-clarity* concepts. For recognition, if there is a circle in the image, it means that *Artinter* recognizes the concept from the image. Bigger the circle is, more confident *Artinter* in recognizing the concept. For the search, I added the CAV of each concept to the vector representation of the query image and drew the nearest neighbors from the WikiArt dataset. 160
A.3 Two cases of how varying weights and scales impacts generation. The scale of “small” has the half of the size presented in “Content,” “Style 1,” and “Style2.” 161

LIST OF TABLES

TABLE

2.1 Background of interviewees. 9

3.1 Reviewed publications according to sampling approaches. 35
3.2 Definitions, example CSTs (e.g.), counts (#), and percentages (%) of codes in each taxonomy. Resource roles are types of supportive resources provided by CSTs, and process roles are about which part of the creation process is supported. Directness is an interaction dimension of whether the user input is close to the artifact or not, while predictability is about whether the tool behavior is predictable or not. Output is about how the tool is contributing to the creation of artifacts. Technologies and users indicate which technologies are used, and who are the intended user population, respectively. 43

5.1 Preliminary analysis results. Q stands for questionnaires and G stands for guidelines. The number in the parenthesis indicates how many questionnaires and guidelines are analyzed. 68

6.1 Participants of the user study. Their domain of interest and years of experience in story writing is shown. 116

7.1 Participants in the user study. Their expertise in visual arts and the domain of interest are shown. We also show their experience using T2I models. 141

ABSTRACT

Artificial Intelligence (AI) and Machine Learning (ML) algorithms are pushing the boundaries of art-making but are hard to direct to the user's intention. They broaden the kinds of artifacts artists can create and can make the art creation process easier and more accessible. For example, prompt-based image generation models provide novel ways to generate images with prompts. Moreover, as prompt-based image generation does not require manual art-making skills, even non-experts in visual arts can experience producing images. However, AI models are often not fully steerable to the user's intention, due to their limited interface and unpredictable behaviors. **In this dissertation, I expand the use of AI models in human art-making by creating effective, efficient, and iterative steering interactions that mix multiple familiar input modalities.** In the first part of the dissertation, I study users and existing creativity support tools to identify how we should design steering interactions for AI-powered tools. From these studies, I found that steerability should let users effectively specify what is allowed and not allowed from AI models with efficient and iterative interactions. Moreover, I identified that artists often convey their intentions about which art to create with modalities beyond those they use to create the artifact. For example, in visual art commissions, they often accompany textual notes along with image references.

Based on the first two studies, I introduce the design of AI-powered art-making tools that mix familiar input modalities into steering interactions. With this approach, I created three AI-powered art-making tools. I designed the first tool, Artinter, to support human-human communication around art commissions. This tool shows how mood boards and concept-based interactions can help in the search and generation of reference artifacts. For example, with Artinter, the user can define their concept of "gloom" in the mood board and use it as a slider to steer search and generation functions to draw more references that can convey the user's artistic intention. TaleBrush is a human-AI story co-creation system powered by generative language models. It allows users to steer generative language models with a visual sketching of the protagonist's fortune. For instance, if the user wants to generate story sentences where the character gradually experiences worse fortunes, the user can visually sketch the fortune arc that goes down. With user sketches, TaleBrush will generate appropriate story sentences. Finally, PromptPaint is a text-to-image generation tool that expands steering interactions beyond text prompts by adopting visual interactions that resemble how we use paint mediums (e.g., oil painting). For example, users can flexibly explore the prompt space beyond what they can verbally describe by mixing the prompts in the prompt palette. Through these tools, my research demonstrates how AI-powered art-making tools can be more usable and useful by mixing familiar input modalities into steering interactions.

CHAPTER 1

Introduction

Advancing artificial intelligence (AI) and machine learning (ML) technologies are changing creative art-making[1], broadening the types of artifacts we can create and the way we create them. Moreover, AI models can make art-making easier with automation and augmentation, allowing users to create beyond their expertise. An example is generative models such as DALL-E2 [330], Stable Diffusion [343], and Imagen [348]. These have caught significant attention in creativity support contexts, as they are capable of generating artifacts with minimal human input. For example, large language models are capable of generating stories or other types of text with few lines of user prompts [44, 329] and diffusion-based models can generate images [330, 348] or videos [373, 407] based on the user's textual prompts. These technologies introduce novel interactions to produce artifacts, while alleviating the required expertise in art-making.

However, many AI models are not fully steerable and configurable to allow "creation" experiences for users. For example, prompt-based image generation models do not allow users to explore artifacts, objects, or styles that cannot be described through language (e.g., how could we describe the mixture of cubist and cyberpunk styles?) or offer fine-grained control of generated results (e.g., make this image have a bit rougher texture). Moreover, AI models might struggle to follow the user's specifications. For example, adding "rough" to the prompt "oil painting" might not result in a version of the image that has a rougher texture than the initial image. Due to limited interfaces and unpredictable AI behaviors, these AI models are not yet fully malleable materials to reflect the user's artistic intentions.

In my dissertation, I show that mixing familiar input modalities into steering interactions can enhance the use of AI models for art-making practices by facilitating effectiveness, efficiency, and iteration. The first part of my dissertation focuses on learning how to design steerability for AI-powered art-making tools. I conducted two studies: 1) studying which interaction patterns users would expect from intelligent art-making support tools and 2) reviewing existing art-making creativity support tools (CST) regarding how they are designed. For the first study

[1] I use the term "art-making" to indicate a broad set of activities for making creative, aesthetic artifacts, from visual arts to music, stage play, visual designs, etc.

(Chapter 2), I conducted an interview study with 14 artists about their experience working with other people, considering other people as *already-intelligent agents*. I expected that people would likely propagate their expectations of already-intelligent agents to artificial ones. For the second study (Chapter 3), I conducted a review of the literature on 111 existing tools, investigating the intersections of roles, interactions, technologies, and users. Based on these studies, in Chapter 4, I identify that steerable AI tools should allow users to specify what is allowed and not allowed from the tool, with iterative and efficient interactions. Moreover, I found that artists often leverage mixed modalities, even those that are not directly relevant to the artifact mediums, to effectively and efficiently convey their intentions.

Based on identified design lessons, I designed and implemented three AI-powered art-making tools that mixed familiar input modalities into effective, efficient, and iterative steering interactions. *Artinter* (Chapter 5) is a visual art communication system that facilitates the search and generation of artifacts with mood-board- and slider-based steering interactions. *TaleBrush* (Chapter 6) is a human-AI story co-creation system that facilitates steerability by allowing users to control the generation by sketching out the fluctuation of the main character's fortune along the sequence of the story. *PromptPaint* (Chapter 7) is a visual image creation system that facilitates the use of diffusion-based text-to-image models by adding traditional paint-medium-like interactions. With these systems, I show that we can design iterative, effective, and efficient steering by surfacing complex vector manipulations into steering interactions that combine input modalities that artists are familiar with.

1.1 AI-powered Art-making Tools

In this section, I provide background and motivation for steerable AI-powered art-making tools. First, I give an overview of CSTs, which include computerized art-making tools (Section 1.1.1). Then, I present an overview of human-AI systems and how AI technologies are impacting the design and use of CSTs (Section 1.1.2).

1.1.1 Creativity Support Tools and AI-powered CSTs

Creativity support tools have been extending human creativity with computational technologies [364, 365]. Art-making, or the creation of artifacts that have aesthetic qualities, has been one of the domains with the active development of CSTs.

While researchers introduced many individual CSTs [119], they also tried to understand their landscape from various points of view. One example focus is the roles of CSTs. For example, Shneidermann [364] described that CSTs can support four types of roles: collect, relate, create, and donate. Similarly, Frich et al. [119] identified the roles that divide the creative process: from pre-ideation to idea generation, implementation, evaluation, and iteration. Another approach was to use metaphorical categories to explain the roles of CSTs [292]: 1) running shoes, for augmenting the artist's creation actions, 2) dumbbells, for helping the artist learn, and 3) skis, for new types of expression. While these efforts identify the range of support that CSTs have provided, they tell less about how new technologies would expand CSTs in their roles and interactions.

What is possible with CSTs is closely related to available technologies. AI/ML technologies have introduced new opportunities for the range of support these tools can provide. Among them, generative algorithms can now even be used to acquire ideas and artifacts that users themselves might not have been able to produce [68, 326]. As AI-powered CSTs (AI-CSTs) with new capabilities can have characteristics different from those built without AI, researchers have begun to study the design patterns of AI-CSTs. For example, the Library of Mixed-Initiative Creative Interface [84, 379] looked into which roles these autonomous CSTs can automate with which types of interactions. Another research investigated more specific AI-CSTs deployed in public spaces and identified design implications for such types [252]. A less common focus is on the model of computer colleagues, where humans improvise with AI-CSTs in real-time with assumptions on more autonomous characteristics in AI-CSTs [77]. Recent advances in generative models further emphasize this aspect of automating generation with minimal human input, such as enabling the generation of images [330, 348] and videos [373, 407] only with textual prompt input. Although they introduce new opportunities to support human creation, more automation has the risk of limiting steerability in human art-making processes. For instance, as many small details in artifacts would be decided by the generative AI models, how can we allow users to take control back in those details? In my dissertation, I investigate design approaches to facilitate steerability in AI-CSTs.

1.1.2 Human-AI Systems

Automation has been one of the core benefits of AI-powered systems, but at the same time, for the task domains where human agency is valued, automation should be balanced with how much agency the user would carry [156, 345, 366, 433]. With generative models, AI-CST designers often try to achieve this balance with steerability or controllability, where humans are steering the high-level behaviors of machines while automating small decisions. There have been many technical control approaches in many mediums, from images [330, 348, 361, 412], to texts [75, 390], audio [134, 178], symbolic music [256, 277], and videos [373, 407]. Their input modalities also varied, from categorical codes [62, 223] to numerical values [256, 434], examples [239, 382], and natural language prompts [187, 330, 348, 373, 407, 424]. However, many steering interactions are often limited in enabling flexible control, like prompts not capable of exploring undescribable concepts. Moreover, many tools have not considered the user's expectations regarding these steering interactions. To identify design approaches for steering interactions, I begin with a study of existing tools and user expectations behind intelligent support.

CHAPTER 2

Artist Support Networks: Implications for Future Creativity Support Tools

2.1 Introduction

While art-making is often treated as an individual activity, the reality is artists often benefit from many relationships in the production of their art. Artists leverage the labor, experience, expertise, vision, and efficiencies inherent in others [25]. These ‘actors’ fulfill many roles: muse, subcontractor, co-producer, critic, mentor, etc. Taken together, these individuals represent the *artist’s ‘support network’*. The artist and the support network ultimately influence the ‘artifact’ to be created, forming a broader *socio-artifact network*—a network composed of interactions between the artist, actors, and the artifact (and its components). In working with the support network, artists cede some power over their process and rely on their human partners. Such relationships are certainly not balanced or uniform in the creation of the artifact. There are extreme variations in the levels of trust, specialization, and power. Additional high-level values such as authenticity, originality, and personal aesthetics also shape the process of art-making. My interest in support networks, and the larger socio-artifact networks, is motivated by the evolution of so-called Creativity Support Tools (CSTs) [118, 292]. CSTs support an extensive range of tasks, everything from coloring technology for black and white images to AI-based generative software that creates scenery, text, or music. As newly introduced CSTs would likely influence the artist’s support network by augmenting and automating the creation process, having a comprehensive understanding of the support network would aid the design of these CSTs.

My goal in this chapter is to understand the unique nature of human-human art practice through a study of artists. Conventional art-making practice is complex and nuanced with a range of dynamics in power. In one extreme, an artist may simply subcontract some tasks, such as color correction or printing a photograph. Here, the artist has “dominant” power–the expectation is whatever they envision or specify will be produced. In contrast, some forms of support are acts of co-creation (e.g., co-authoring a book). These relationships would have a different set of dynamics, with more equal power structures. Disputes would require the artists to come to an agreement on what to create. In between these extremes, we might have a case where a movie director (one artist) collaborates with a music composer (another artist). The director may let the composer decide on many details or even the high-level direction of the score. However, even between these two artists, there are unequal power dynamics. The composer will often have the creative power of a limited facet (the score) of the completed artifact (the film). These are but a sample of the relationships that exist in art-making.

To understand the broader spectrum of human-human support, I conducted an interview study with 14 practicing artists from fields ranging from visual arts to music, creative writing, and game design (with a mean experience of over 9 years). Through semi-structured interviews, I focused on the artists' experience of working with others in their efforts to create art. Though I target broad artistic domains, I identify common patterns in the roles, dynamics, and success (and failure) stories across these interviews.

The relationships I observed can be viewed as a type of network centered on the artist. Elements of this network shape the specific piece of art and are often the targets of software designers (in both academic and industry). Perceptions of 'inefficiencies' lead to the design of software that automates or augments parts of the network. However, in art-making specifically, a tool solely focused on improving efficiencies may not be desirable. Based on my model of how human actors play their roles in the artist's support network, other aspects, such as power dynamics, ownership, or trust, need to be considered by designers. I hypothesize that CSTs that follow support patterns familiar to artists would be more likely to be accepted. Through this lens, I offer several insights into how a subset of CSTs, those that support human-human collaboration (collaborative CSTs) and artificial intelligence-powered ones (AI-CSTs), can be better integrated.

This chapter contributes a novel perspective of human-human relationships in artistic contexts. This expands on past efforts, which were often explicitly grounded on technically-focused collaborations. For example, sites of prior work included collaborations between technologists and artists [267] or systems that support online-collaboration [261, 262, 263, 357]. From a wide spectrum of art-making domains, my analysis finds an array of relationship types ranging from *subcontracting* relationships to *co-creation* to *mentorship*. Through interviews, I identified specific mechanisms of how these relationships work—higher-level types of support (i.e., those that impact the art and those that impact the artist or process) and dynamics (i.e., whether the supporting actor can make artistic decisions). Finally, my analysis reveals patterns that lead to successful relationships. Using this analysis as a lens, I offer several insights regarding the design of steerable AI-powered art-making tools. Artists often work with supporting actors with iterations—they iteratively communicate their intents and gradually co-build the artifacts by passing those between the artists and supporting actors. Within iterative work, artists tend to receive two types of support on the artifact: implementation and ideation. Moreover, artists and supporting actors tend to use communication means of multiple modalities from verbal specifications to references and sketches. This chapter is based on the paper “Artist Support Networks: Implications for Future Creativity Support Tools,” published in DIS2022 [65].

2.2 Background and Related Work

Social environments influence creative actors in numerous ways [8, 9, 18, 110], both indirectly and directly. Indirect influence can include one artist influencing another stylistically [90], or art collectors shaping a domain through market forces [191]. Other actors more directly influence the artist and their creative process. The combination of direct and indirect forms parts of what Becker calls the “art world” [25]. My goal is to understand those interactions in which an artist can request support. Thus, I restrict the definition of the artist’s support network to only include those humans that have an aligned goal or in which the artist interacts intentionally and directly. I exclude elements that may have influenced the artist but not intentionally (e.g., Mozart’s influence on artists 200 years later).

2.2.1 Human Support for Art Creation

The idea that art-making is social is not new. Significant research has identified the various social, psychological, and economic processes in which an artist is embedded [24, 25, 190, 350]. In this broad “art world” [24, 25], there is a range of roles–everything from muse to mentor to collaborative partners to collectors to critics. Within this broad definition, Becker identified the “collective” process of art-making by which actors provide support in art-making through “division of tasks (or labor)” [24, 25]. In some cases, these actors may reflect specialists to which the artist allocates work (e.g., a color editor or music producer). However, these agents need not be “subordinates.” For example, an artist’s support network can include peers (e.g., co-authors, co-composers, etc.). The division of tasks in this context is often a cooperative allocation. Collective activities are natural for domains that involve many people in the creation, such as movies or stage plays [24, 25, 350]. However, even for those forms of art that are traditionally considered ‘solitary’, we can find support networks. For example, a poet may benefit from editors and even type-setters who can help shape the produced work.

There have been efforts to create finer-grained taxonomies within the HCI community around human support in art-making. The computational aspect in these taxonomies has often restricted these studies to technological contexts (i.e., artists working with technologists or technology as an intermediary between artists). However, some findings can be generalized to broader contexts. For example, Mamykina et al. [267] focused on types of provided support in the context of co-work between artists and technologists. The work identified three types of support: 1) creative concepts like the core ideas and visions about the art piece; 2) construction, like the execution of the artifact; and 3) evaluation, inspecting if the creation is done according to the vision. Within HCI, we also find targeted work on situations in which human collaborations are successful or can be enhanced. These include settings varying from online collaboration [262, 263, 357] to the co-work of artists and technologists [267, 445]. Generalizable findings within this research have identified lessons that lead to successful collaborations (e.g., common vision, knowledge sharing, systems that facilitate communication, or early planning). Compared to these efforts, I take a more general approach to understanding an artist's support network. First, I am interested in those common features that exist across a broad set of artistic domains. Second, I include support relationships that are not necessarily cooperative or collaborative. These can include a muse or mentor or even a commissioning agent. As long as these individuals broadly '*intend*' to influence the art or art-making process, and the artist *intends* to get support from them, I consider them as a part of the artist's support network. Notably, I expand beyond artifact-centered relationships. That is, I also consider relationships that can impact the artist *directly* but the artifact *indirectly*. For example, I include those relationships that shape the artist (e.g., mentorship) rather than the art.

Again, I have selected this area of focus as the individuals or roles involved are possible targets for CSTs. While this chapter's interviews are only loosely structured, the probe questions and analysis were focused on identifying and classifying these types of roles and interaction dynamics.

2.3 Research Method

Rather than focusing on a single type of artist or art-making practice, I explicitly sought to interview across domains–visual arts, music, creative writing, etc. My goal was to identify patterns and differences both within and between artistic domains. By focusing on a broader set of domains, I identify the wider range of facets of an artist's support network. Though I recruited professional artists for my interviews, I tried to vary seniority levels among participants. My hypothesis was that different seniority levels interacted with different types of support networks. I utilized an approach that mixed theoretical [289] and snowball sampling [140]. I strategically invited interview participants based on the intermediate results, focusing on interviewees who could offer alternative perspectives (e.g., by varying the domain, experience levels, etc.).

	Domain	Experience	Current Occupation	Examples of Created Artifacts
I1	New Music	24 years	Composer	Commissioned compositions; Personal compositions
		2 years	Harpsichord player	Accompaniment for other performers
I2	Classical Music	10 years	Composer	Music for musicals, plays, and other performances
I3	Metal Music	8 years	Composer	Metal songs and albums
		15 years	Guitarist	Metal stage performances; Featuring on songs of other artists
I4	Metal Music	18 years	Composer	Metal songs and albums
		22 years	Guitarist	Metal stage performances
	Screen Printing	4 years	T-shirt printer	T-shirts for own and other bands
I5	Choral Music	+20 years	Composer	Commissioned compositions for choirs; Personal projects for performances
	Showbiz Music	23 years	Music Editor	Editing on TV or film music and sound
I6	Video Exhibition	5 years	Video/Music Creator	Commissioned exhibitions for galleries
	Indie Electronic Music	7 years	Composer/Performer	Electronic music songs and albums; Electronic music performances
I7	Visual/UX Design	3 years	Visual/UX Designer	Visual designs for software applications
I8	Visual/3D Design	2 years	Visual/3D Designer	Visual designs for software applications; 3D modeling work for other artist's project
I9	Independent Animation	3 years	Animator	Independent animations for film festivals
	Visual Design	8 months		Visual designs for book covers; Animated emoticons
I10	Video Art	1 year	Fine Artist	Arts for exhibitions, including drone video arts
	Exhibition Art			
I11	Visual Art	2 years	Visual Artist	Toreutics (metalworking); Visual arts for exhibitions; An illustration book
I12	Games Art	3 years	Games Artist	Graphic assets in games
	YouTube	3 years	YouTube Creator	YouTube videos
I13	Game Design	3 years	Game Designer	Mechanics and level designs in game
			Game Developer	Programming and implementation of game
I14	Creative Writing	12 years	Creative Writer	Novels; Interactive novels
	Indie Pop Music	6 year	Composer	Indie pop songs and albums
		12 years	Guitarist/Vocal	Indie pop performances

Table 2.1: Background of interviewees.

2.3.1 Participants

Table 2.1 details the 14 artists who participated in interviews. As some artists worked in multiple mediums, participated interviewees covered 20 unique domains. These range from music to visual arts, visual design, exhibition art, game design, and creative writing. All interviewees were currently active in at least one domain and had one or more years of experience in their 'primary' domain (experience across domains ranged from 8 months to 24 years). While I did not pre-filter interviews based on collaboration experience, all interviewees had worked with others for some part of their creative workflow.

2.3.2 Interview Protocols and Data Generation

I conducted semi-structured remote interviews (a mix of audio and video calls based on the interviewee's preference). On average, interviews lasted an hour and ranged between 35 and 100 minutes. Interviewees were paid with a \$20 gift card. Interviews were recorded and transcribed with additional notes taken by the interviewer during the call. Interviews were analyzed using constructivist grounded theory (detailed below).

While I allowed my interviewees to largely direct their focus, I specifically probed human relationships as part of the artist's networks. Specifically, I was interested in the dynamics that artists have with other people[1]. I specifically asked about motivations, mechanisms, successes, and failures from support experiences.

While I primarily focused on situations when the artist was 'central', participants were also free to describe occasions when they fulfilled supportive roles *for* others (e.g., when they acted as a sub-contractor). I consider both dynamics important. As I describe below, the notion of centrality in this network, or of being the 'main' artist, relates to power dynamics. One can view the artist support network as an egocentric social network centered on the *main artist* (i.e., the "ego" in an ego-network). In my definition, the main artist carries more decision power and gets the majority credit for the art piece (often receiving top billing or 'solo' credit). To simplify my naming, I will use *actor* to refer to those people in the artist's support network (these are referred to as the "alters" in an ego-network). In the pair of *main artist* and *actor*, the actors most often have less power, less creative control, and less credit (though in some rarer cases both individuals may have equal power or credit). I limit actors to those individuals who have some bi-directional interaction with the main artist. This excludes people who might be an influence on the main artist but are in a different artistic generation and have never interacted. In this model, it is often the main artist who 'activates' the engagement with the actor. While I use the simplified 'artist,' I note that some

[1]Note that I did not use the phrase 'support network' during the interview. I did not want my participants to fixate on the idea of a social network or only those individuals that met some restricted model of 'support'.

individuals did not regard themselves as such, preferring *designer*, *creative*, etc.

2.3.3 Analysis Method

I used constructivist grounded theory [137] to analyze the interviews and derive a theoretical model that can explain the artist's support network. While traditional approaches of grounded theory allow little prior knowledge [136], the constructivist version allows researchers to bring in prior perspectives when making interpretations to construct theories [385]. Adopting constructivist grounded theory, I considered prior work on support provided by actors when conducting data collection and analysis. I also considered my scope of support network and actors that I discussed above. While the prior knowledge of CST systems likely influenced the interview analysis, I did not use taxonomies from that space explicitly. Instead, I focused on those roles and dynamics highlighted by the participants from their existing support networks. For the analysis, I and another author conducted the theoretical coding. Specifically, we conducted open coding, axial coding, and selective coding in different stages. Different artist comments were first placed on cards. We grouped these comments according to each art-making experience that artists had with actors. Within each experience, the authors collaboratively performed low-level coding. Different experiences were compared, and those with similar codes were merged, forming categories of experiences. These categories resulted in a set of relationship *types* within an artist's support network. Relationships that only exist in a single domain have been excluded as we aimed to identify support patterns general across art domains. Through analysis, we identified patterns of interaction shared between relationship types (e.g., similarities between subcontractors and featured artists). From these, we determined higher-level codes about provided support and success conditions. We conducted this process while continuing our interviews, testing intermediate theories against new data [289]. We noted patterns of interactions between artists, actors, artifacts, and provided support for different support relationship types. As I describe in a later section, these interactions could be modeled as a socio-artifact network (i.e., the people and artifact components) with a 'support network' at its core (the human social network). As such, taking inspirations from social-technical software network [403], I decided to model these interactions in graphical format (see Figure 2.2). I derived the specific format based on the analysis.

2.4 Results

The analysis revealed a range of relationships between artists and members of their support network. Within these relationships, I focused on four broad facets: 1) The specific 'support' provided within the relationship (e.g., implementation, ideas, management, etc.); 2) the relationship types

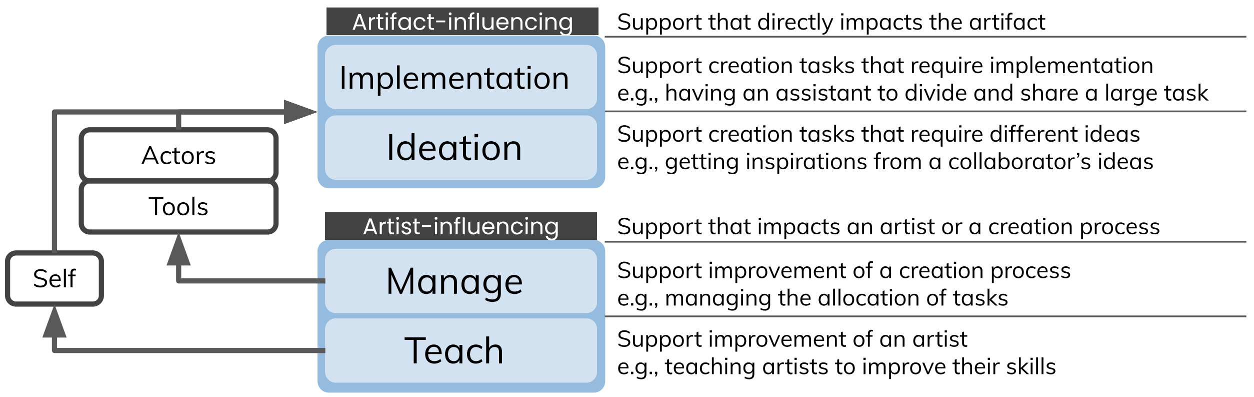

Artifact-influencing
Support that directly impacts the artifact
Implementation
Support creation tasks that require implementation
e.g., having an assistant to divide and share a large task
Ideation
Support creation tasks that require different ideas
e.g., getting inspirations from a collaborator's ideas
Actors
Tools
Self
Artist-influencing
Support that impacts an artist or a creation process
Manage
Support improvement of a creation process
e.g., managing the allocation of tasks
Teach
Support improvement of an artist
e.g., teaching artists to improve their skills

Figure 2.1: A summary of the support types identified through our study. Artifact-influencing support directly contributes to artifacts, while artist-influencing support impacts the artist or the creation process. The arrows on the left indicate possible feedback and controls. Though the dynamics vary, artifact-influencing support (i.e., implementation and ideation) are influenced by the artist (self), their collaborators (actors), or tools around artists and the actors. Among the artist-influencing support, managing impacts creation through the management of actors and tools (a meta-role), while teaching impacts by artist's self-improvement.

(i.e., real-world categories, like 'sub-contractor,' which are often determined by the artistic community); 3) the dynamics of the interaction (e.g., power dynamics, ownership, etc.); and 4) the conditions for success. While taxonomized separately, these facets interact in complex ways. Clearly, any particular relationship can consist of multiple types with a spectrum of provided support and/or dynamics. For example, one type of relationship (e.g., 'featuring'[2]) could provide multiple types of support (e.g., providing skills *and* ideas–in this case, lyrics and vocal support).

2.4.1 Support Provided in the Artist's Support Network

Participants described various supports provided when interacting with other actors. These fall into two main categories: *artifact-influencing support* and *artist-influencing support*. *Artifact-influencing support* impacts the art piece itself and include *implementations* and *ideas*. An alternative way to view these specific categories is from the artist's perspective. They are things that are often limited or exhaustible for the artist when creating an art piece. I contrast these to *artist-influencing support* which includes *managing* and *teaching*. *Artist-influencing support* can be viewed as more 'meta' as it influences the properties of the artist (e.g., enhancing their skills or teaching them new techniques) or impacts the artistic creation process (e.g., managing the relationship and interaction among subcontractors). Within each of these categories, I observed a few specific sub-categories based on what form the influence actually took. Actors sometimes provided

[2]In music, 'featured' artists are often those providing vocal or instrumental support to a track created by another artist. A featured artist may be famous in their own right.

support in the combination of types (e.g., implementing while also giving some ideas). Figure 2.1 summarizes my high-level categories.

2.4.1.1 Artifact-influencing: Implementation

Artists indicated that when they could not do an *implementation* task due to a lack of resources, they would recruit from their support network. Participants mentioned two types of specific *implementation* support: *expertise* and *labor*.

First, artists leveraged support from others when they lack specific, specialized expertise. When expertise-wise support was required, often, my participants were involved in interdisciplinary efforts, such as games or videos. For example, when I10 needed to use a drone to take a video of a drawing on the beach, they drew on help from a specialist who *"had more experience working with a drone"*.

Even when an artist themselves possesses the *expertise* to do something, they might lack the time or other efficiencies (i.e., a comparative advantage). In this case, the *expertise* becomes a substitute to the artist's own skills. I refer to these relationships as providing *labor* support. In many situations, artists wanted other people to do a task if it required a lot of human effort. My interviewees indicated that this happened frequently in industrial settings. I12, a game artist, noted: *"I guess regarding, specifically for collaboration, in the industry, collaboration is absolutely necessary just because, the amount of work, like, there is no way you could get through everything by yourself."*

2.4.1.2 Artifact-influencing: Ideation

In interviews, *ideation* was often described as support that was distinct from *implementation*. Artists described *ideation* as providing exposure to novel ideas and generating *inspiration*. For instance, I14 described how their close exchange of ideas with others can lead I14 to reach *"interesting junctions"* and *"surprising things"*, that I14 *"wouldn't do normally"*. Related support leverages others' opinions as a type of critic to provide *feedback* on a specific artifact. For example, I13, who is working in a game company, used in-company feedback to decide which parts of the game needed to be improved most quickly, and which projects were worth continuing. The relation between *ideation* and *implementation* would seem analogous to that of supporting designer's 'thinking' and artifact-wise 'outcome' from Stolterman and Selvan [384]'s work on designerly support. While Stolterman and Selvan considered that 'thinking' does not contribute to the artifact outcome, I took a different perspective that 'ideation' also contributes to the formation of the artifact.

2.4.1.3 Artist-influencing: Managing

The *managing* support category is often centered on organizational help. For example, I13 noted how the CEO in their game company makes major decisions about who works in which team. The CEO was not necessarily the main game designer (the central artist in this scenario). However, they helped organize the artist's support network to execute the vision. This may be in dividing, or enabling the division of providing of labor, expertise, and ideation to certain individuals in the network. Those who provide *managing* support can also recruit other people or acquire tools, expanding the artist's network. For example, I9 mentioned the role of an animation production studio as follows: *"They support me with things that I need. For example, . . . I ask if they can support me with the creation of the file, then the producer says like 'oh I can try to find a person who can work on that in the studio"'.*

2.4.1.4 Artist-influencing: Teaching

The second *artist-influencing support* is *teaching*. This support was often in the form of educating the artist on a new technique, tool, or style. As the result, artists may gain *expertise*, efficiency for *labor*, and *ideas*. Artists described that learning would sometimes happen through demonstration. For example, I5 recalled learning composing techniques from a mentor, who demonstrated how various styles could be applied to one theme.

2.4.2 Relationships and Dynamics

Having defined the provided support, I now introduce the cross-cutting *relationship* types in the artist's support network (Figure 2.3). Some of these were the formal names or job titles of the people that formed these relationships, which were used for crediting purposes. For example, a 'featuring' relationship has a specific meaning in the context of music production (another artist that acts as a guest performer). These names often have a constructed, or even legal, meaning to different artistic communities (e.g., 'compilation' is something different in music and literature). As such, I highlight the different dynamics that emerged in interviews with artists when reflecting on relationships.

I attempt to graphically represent each relationship type (Figure 2.2): which types of support are provided by whom (artist or actor); how much is each individual providing (and in what form– idea or implementation) towards the artifact; and is the support directed at the artist or artifact. The pieces of this socio-artifact network were partially inspired by social-technical software network [403]. My models include the artist and the support network (composed of actors) as PEOPLE nodes. They also include ARTIFACTS to be created. These models indicate support relationships with the PEOPLE nodes and edges from them. Artists and actors are connected with edges based

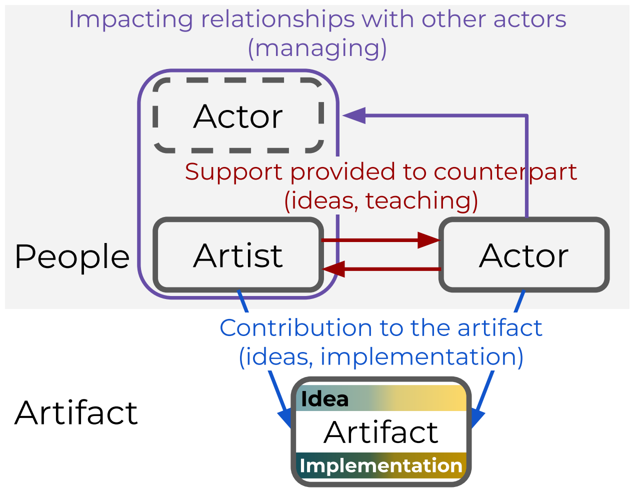


Figure 2.2: I describe socio-artifact networks with relational graphs of PEOPLE and ARTIFACTS. Artists and actors provide the support that impacts artifacts (ideas and implementation, with blue edges) or each other (ideas and teaching, with red edges). Actors also can provide 'managing' support to the artist and other actors (purple edges). The amount of an idea or implementation each person provided to the artifact is color-coded in the artifact diagram. The contributions made by the artist are in light teal for ideas and dark teal for implementation. For contributions from the actor, I use light yellow for ideas and dark yellow for implementation. The gradient indicates a 'range' of values.

on which types of SUPPORT are provided from one to the counterpart (red and purple edges in Figure 2.2). Red edges stand for providing *ideation* or *teaching* to the counterpart. On the other hand, purple edges indicate *managing* support. Note that the artist would also provide SUPPORT to the actor, in order for actors to accomplish SUPPORT that the artist wants.

ARTIFACTS are usually of one type. Though I could conceivably break the artifact into subpieces—as one might do in a socio-technical model—I did not find this kind of modularity informative (e.g., how does one modularize a printed photograph? or music recording?). The only exception is when multiple ARTIFACTS from multiple artists aggregate to form a meta-artifact, an idea I return to. ARTIFACT nodes are connected to PEOPLE according to the direct contributions to the creation of the artifact, which can be *ideation* or *implementation* (blue edges in the figure). The blue edge from actors represents SUPPORT to the artist. The width of edges indicates the amount of support or contribution made from the tail of the edge to the head. The colored glyphs inside ARTIFACTS nodes indicate how much of the *ideation* or *implementation* was contributed by the artist or actor. Teal indicates contributions from the artist and yellow indicates those from the actor. These colors are expressed in a gradient in glyphs to show that there can be a range of values in contributions. For example, subcontracting may include more or fewer changes to the artifact based on the kind of subcontract. While I focus on describing relationships between one

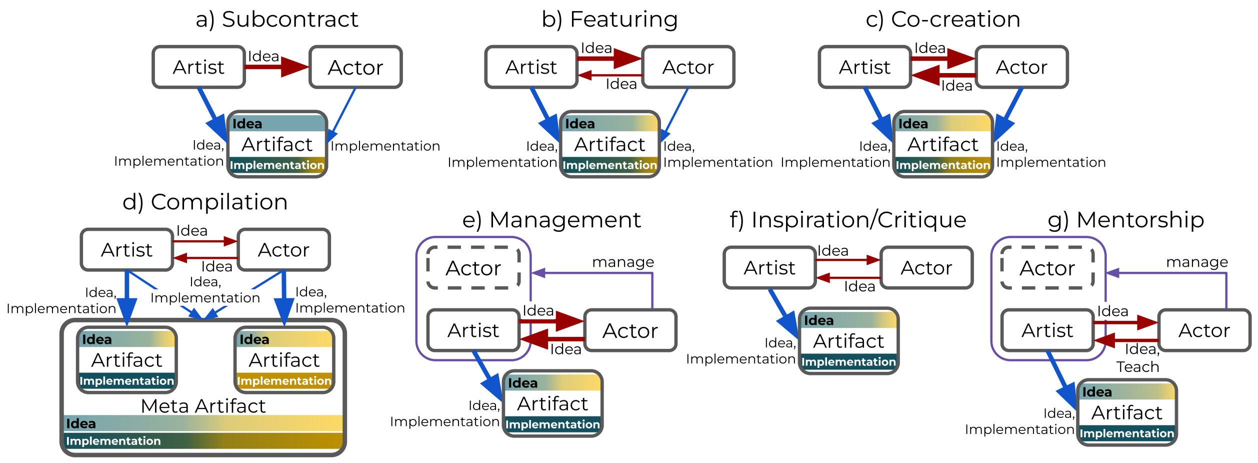


Figure 2.3: Diagram of relationships in support network. Support relationship types are described as relational graphs between artists, actors, and artifacts. The width of edges indicates the amount of provided support. The amount of idea- and implementation-wise contributions made by the artist and the actor to the artifact is denoted with gradient glyphs.

artist and one actor, multi-actor situations can be described with the combination of one-on-one relationships. While I identified the sense of ownership felt by the artist or the actor, I did not indicate it in the diagram. Note that the concept of ownership is relevant to other artistic values such as authenticity [296] and authorship [276].

2.4.2.1 Subcontract

A *subcontract* is a relationship where the main artist has a 'target' artistic idea, but needs *implementation* support such as *expertise* or *labor* to realize it (Figure 2.3a). Subcontractors are often specialists (e.g., a film editor), but they need not be. In subcontract interactions, artists convey their ideas about what to create to actors, and the actors directly implement the requested part of the artifact. From the actor's (i.e., the subcontractor's) perspective, they are given a specific and separable task. For example, I11 described the experience of asking material shops to do a task for toreutic (metalworking) pieces: *"To a certain stage, I do things by myself, then, I bring those and explain to people there [material shops], things like, 'I need this, this, and this, so please work on these processes'."* In the final artifact, only a small portion of the implementation is perceived as coming from the actor. More critically, they are not perceived as contributing much to the idea/vision of the artifact. While creative freedom is limited for the actors, sometimes, they provide feedback to the artist (e.g., technical feasibility). It is particularly true when the actor may have more expertise than the artist. I11 reflected: *"There are cases where I thought the thing would work, but the thing actually does not work from the engineer's perspective . . . If they tell me technical conditions like 'this should be done in this way', I try to fit into those conditions, or I revise my own plan."* In this support relationship, the actors tend to have a weak sense of *ownership*. For example, I9 did not feel a sense of ownership when they participated as a subcontractor on a specific project. The participant reflected on how they had no freedom and could be replaced by anyone with similar expertise.

2.4.2.2 Featuring

In *featuring* relationships (Figure 2.3b), the main artist asks the actors (often another artist) to create their own part within the main artist's piece. The artist anticipates (and expects) actors to create elements that artists cannot due to lack of *expertise* or differences in styles and *ideas*. Unlike subcontracts, featuring relationships mean that while the artist leads the overall idea and implementation of the artifact, actors will be given a small section of the artifact where they can exert their ideas and implementations. In the cases where the 'guest artist' (i.e., the featured actor) might be more famous than the 'main artist', the actor's power may be significant. In featuring relationships, the main artist often benefited by dividing *labor*. While the idea of featuring, or guest artists, is most common in music, there are analogous relationships in other domains. For example, I8 shared the experience of working in a web-application company as a graphic designer: *"The overall plan and design are done first, like, UI designers and directors do that first. . . . For decisions like 'where to put which graphics', I do not carry much decision powers. However, in terms of 'how to express the graphic in which way', I could make more decisions."* I note that the specific name ('featured') was rarely used outside of music. Additionally, I found that analogous relationships tended to provide less freedom to actors in non-music domains. However, a key distinction of this relationship that was both common across domains and also distinguished it from sub-contracting was that the actor's ideas tended to influence and shape the 'vision' of the final artifact. While the actors did not contribute to the whole art piece, they did retain some *ownership*, at least for parts they contributed. For example, I12 noted the attachment towards game artwork on which I12 contributed core ideas and manual efforts, even though that was a small portion of the final game.

2.4.2.3 Co-creation

Co-creation is a type of support relationship where there is little to no distinction between the main artist and the actor (Figure 2.3c). Artists involved in *co-creation* thought of it as a form of close collaboration. They collaborate for many reasons, including complementing each other's *expertise*, dividing *labor*, and exchanging *different ideas*. All team members contribute to the ideas and implementation of the artifact, to a similar amount. Hence, all members tend to work closely throughout the whole creation process. This is not to say that team members don't sub-divide the work or focus areas (due to differing levels of expertise, experience, interest, etc.). As artists and actors work closely, they actively communicate ideas to each other. For example, I14 mentioned the experience of writing a story on an interactive art that includes 3D modeling of fairytale characters: *"So, it begins with talking with them [3D modelers] about how they're thinking about the project . . . And then once you see the model, you kind of see like, how aesthetically they're thinking about the creature that we picked . . . Then, you can start to write a story that will fit the aesthetic that's present. So it goes back and forth like that. So, one person makes something, one person explains it, they try to fit and then they adjust them until they come together better."* Artists generally mentioned that they shared *ownership* with team members. It was sometimes hard to draw a specific line that distinguished if a relationship was a subcontract, featuring, or co-creation. For example, just how much of the artifact's final vision came from the actor to make 'featuring' into 'co-creation? Though in music, these may be resolved through a legal definition, this is not always the case. An alternative view is that even though the extremes are obvious, these three types can be viewed on a spectrum with fuzzy boundaries.

2.4.2.4 Compilation

Compilation is another relationship type without a clear distinction between the main artist and the actor (Figure 2.3d). Artists also thought of this support relationship as a close collaboration. In *compilation*, artists with similar *ideas* or *styles* gather together and present their creations in one *meta-artifact*, which includes all artists' artifacts, such as an exhibition or performance. In this relationship, artists can exchange *ideas* to agree upon the topic of the meta-artifact or how to structure it. While each artist work on their own artifact, they can collaboratively contribute their *ideas* and *implementation* efforts to weave individual artifacts into one meta-artifact. On each individual artifact included in the meta-artifact, the artist would receive idea "contributions" from the actor while the artist would implement their own artifact. For example, I10 exhibited a piece with another artist, only collaborating for *"figuring out the common theme for the show"* or exchanging ideas, but not for the artwork itself. A meta-artifact, on the other hand, would receive a similar amount of idea- and implementation-centered contributions from all involved. As each piece in the meta-artifact is presented under the name of a single artist, the sense of *ownership* for their specific piece would be high.

2.4.2.5 Management

In *management* relationships, the main artist gets *managerial* support from the actor (Figure 2.3e). For example, I9, who is an independent animator, got support from a producer to raise funds and manage necessary human resources. Note that *managing* in Section 2.4.1.3 is a category for a type of support, while *management* is a type of relationship that include *managing* as one type of provided support. As such, actors in management roles might also make *idea*-focused contributions. This contribution from the management actor usually made a significant impact on the artifact. For example, I9 described the experience of working with the animation producer. I9 had *"the leading power"*, but *"had to also consider the producer's interest, taste, and options"*. The amount of support would decide how much power actors carry. However, the actor would rarely make direct contributions like implementing the artifact. The significant contribution on *ideas* also affected the sense of *ownership*, the actor taking some portion of it from the main artist.

2.4.2.6 Inspiration/Critique

Support also happens when the main artist seeks *inspiration* or *critique* from others (Figure 2.3f). *Inspirations* and *critiques* can happen by the actor giving *ideas* to the artist. For example, I1 noted that exchanging *ideas* with other composers could be helpful as they have *ideas* on *"totally different directions, from which you can get benefits"*. In this support type, main artists can decide whether to accept provided ideas or not, while actors would have freedom on which ideas to give. For example, I13 mentioned that when analyzing feedback on developed games, as there are *"many types of users"*, I13 tries to identify from which types of users the feedback is helpful. As the actor does not directly contribute to the artifact and as the artist makes decisions on whether to accept the actor's ideas, the actor would have almost no sense of *ownership* of the main artist's piece. Unlike inspiration, critique requires the actor to observe the artist's artifact, as the actor needs to give feedback on it. Note that, in these interactions, we only considered cases where the actor intended to help the main artist (and the artist solicited this advice). This is in contrast to a newspaper critic. In that situation, the critic is providing unsolicited feedback. Moreover, the target of their advice is not the artist specifically, but their readers, generally.

2.4.2.7 Mentorship

Mentorship (Figure 2.3g) is the support relationship type where the main artist seeks skill-, career-, and creation-wise development from the actor, or the mentor. The support usually happens in the form of transferring knowledge (*teach*) or giving feedback and suggestions on directions (*ideation*). For this support type, specific dynamics can differ a lot based on how much authority the actor has. For example, I1 mentioned that in classical music performance, as *"there are more standardized styles and conventions"*, mentors would tend to *"have the authority"* to give feedback to students. On the other hand, for cases where peer colleagues do a single-time session to teach each other, due to less authority from peers, the actor's power on the creation would be much smaller. In this relationship, the actor can contribute ideas (to a varying degree based on the authority the actor has), but not implementation-wise contributions to the artifact. As the powers of the main artist and the actor largely depend on the extrinsic factors, the sense of *ownership* would also depend on it. In some cases, through mentorship, artists shift the ‘bundle’ of people and tools in their network, which is a type of *managing* support. For example, I4 learned T-shirt printing by attending a local workshop on screen printing. This mentorship experience might lead to new people and tools being integrated into the support network. For example, having learned screen printing, the artist might want to recruit help to produce multiple versions of the shirt. Unlike a critique interaction, a mentorship is often long-lasting. However, as with subcontracting/featuring/co-creation, these relationships may fall on a spectrum.

2.4.3 Frictions and Conditions for Success

Given the various combinations of supports, interviews revealed various frictions and solutions that made these relationships fail or work. I found that frictions were largely due to two categories 1) having different styles, values, or levels of knowledge, and 2) external factors, such as social dynamics.

2.4.3.1 Artistic Frictions: Different Artistic Values and Knowledge

Artists often have different artistic styles, expertise, ideas, and perspectives. While an actor’s diverging perspective can be inspirational, interviewees noted that it also can be the source of disagreement and blur an artist’s own vision. For example, I7, a visual designer, mentioned the difficulty in collaborating with another designer: *“If we discuss ideas only based on our own design styles and tastes, the process of giving feedback fundamentally becomes subjective. Furthermore, if we have some sort of pride in our own designs, it would be harder to exchange opinions.”* Similarly, I11 noted critiques from diverging perspectives can negatively affect a work: *“When there are too many people giving critiques, my direction can change to an unintended way. Like, I assumed that a certain thing is important, but there can be people who mention other things as important. So, such things can be a difficult case.”* My interviewees suggest a number of approaches for this friction type.

Trust in Understanding Styles and Values A common theme described by artists to resolve friction was building trust and understanding of each other’s styles and interests. Styles and values underlying art pieces are crucial to the artist, as they are closely relevant to one’s originality and personal aesthetics. Artists often build this shared understanding by having iterative and repeated co-work experiences. For example, after doing some rounds of collaboration with a friend, I10 *“knew that it would be super easy to collaborate”*. Sometimes, artists described converging on a similar taste or color. This approach was mostly mentioned for support relationships where the main artist and the actor exchange ideas about the artifact for a extended duration of co-work (*featuring*, *co-creation*, *compilation*, *management*).

Trust in Skills and Quality Artists mentioned that having an actor with better skills and more experience would sometimes positively influence the interaction. In particular, familiarity with the other artist's work helped to manage expectations. For example, I5 indicated that when a chorus gives commission to composers, they prefer those with known skills and experience. That is, they would not hire someone who does not understand *"technical problems"*, or would write a piece that *"cannot be sung"*. This solution was frequently brought up for relationships where the actors directly contribute or impact the artist's artifact (*sub-contract*, *featuring*, *co-creation*).

Communication Most support types benefited from iterative communication. Through effective communication, artists could define and scope the relationship and reach agreements. For instance, I7 indicated that when disagreements arise within a team, the team *"had an open discussion and listened to others, to reach the agreement"*. However, there could still be difficulties in communicating the direction of art: the ambiguity of communication means. While ambiguous verbal languages can be an effective means to convey rough, high-level ideas, with them, people could also perceive one idea differently. While artists can *use jargon* to overcome this challenge, they would not be able to express all ideas. Moreover, jargon is limiting when artists from various fields work together. For example, I2 noted the difficulty of communicating musical ideas: *"When musicians are talking to each other, 'Increase half key', like this, we communicate with musical languages . . . However, apart from that, we frequently use adjectives to explain things. Like, 'let's go a bit shy', 'this part should be raving', or, 'let's go with tight sound'. Then, everyone somehow has some ambiguous images about these words . . . If collaboration happens with other genres, I think this problem can be even bigger."*

One approach to overcome the challenges of ambiguous language is to *use references* or *sketch ideas*. For example, I3 mentioned that when recording a guitar track for others, I3 would *"record as many things as possible"* to reach the agreement. If it did not work well, I3 and collaborators would *"bring references"* and *"play as much similar as those [sic]"*. For this approach, iteratively using slightly different versions of references and sketches could help, as such small differences between those versions would highlight the directions that the artist and the actor can agree upon. However, these approaches can be difficult when the cost of finding references or creating sketches is highly significant. I12 described a situation in which the game company asked people to implement many ideas about the game arts. This led to *"so many extra artworks generated but never get used in the final game"*. Furthermore, references can still be perceived ambiguously. For instance, I2 mentioned the experience when a movie director shared a video as a reference for composing background music: *"What the person got was more about the combined image of all factors within the video... I would only refer to the music of it, and I would create a music piece based on it. But when the reference is separated in such a way, it can be perceived in a very different way."*

Concentrating Power A solution to frictions caused by value and style differences is to concentrate decision power on one artist. One mechanism is to allow the main artist to explicitly filter decisions and *selectively accept other ideas* based on their own criteria. This approach is common when there is a significant *critique* or *inspiration* by a collaborator. For instance, when I13 was designing a casual game for a broad set of the audience, I13 did not consider feedback from a hardcore gamer as critical.

Another mechanism for concentrating power is *dictating*. With this approach, one artist would give clear specifications and highly constrained authority to actors. For example, I9 mentioned that when asking others for manual tasks, I9 would hand out *"a method sheet"*, or *"a manual"* about how the task needs to be done. This specific approach would work for tasks that do not require creativity from actors (*subcontract*). Based on how much authority each artist has, the level of control that can be imposed on the support relationship would differ. For example, I1 stated that in the classical music domain, conductors' power to performers is *"absolute"* and they usually *"carry all the power"* about the interpretation of the song, and thus have ultimate say on what each performer does.

A final alternative is to simply relinquish all decision-making power to the actors. This approach would be most effective when the actor contribute both ideas and implementation on a co-created artifact (*featuring*, *co-creation*). Artists tended to use this approach when they had clearly different expertise or tasks. Power can be distributed in such a way as to reduce overlap. For example, I6's exhibition team had few conflicts as *"the separation of the roles is quite clear"*. It is important to recognize that moving power–either to the main artist or to the actors–may often make matters worse when working with other people.

2.4.3.2 Extrinsic Frictions: Additional Factors

Some frictions emerged from extrinsic factors. I identified three types of non-artistic frictions: 1) unfair relationship, 2) insufficient resources, and 3) personality. While interviewees mentioned problems caused by non-artistic factors, they did not mention many solutions for them.

First, *unfair relationship* can lead to imbalanced work-credit allocation (too much work, too little credit; too little work, too much credit). For example, I9 reflected that freelance visual designers are vulnerable to unfair contracts, which do not specify *"how many rounds of iterations are allowed"* for the requester. In these situations, freelancers would often overwork, as requesters might ask for revisions until they are satisfied. Likewise, I12 mentioned that in the game industry, even though artists do *"a lot of things in the company"*, only *"the tip of the iceberg"* would be released, and artists would not have clear credit for what they did.

Second, *lack of resources from actors* also causes problems. An actor might not be able to invest enough resources to help the main artist. For example, I14 mentioned he did not get much help from his mentor in the educational program as the mentor was *"busy"* and had *"a lot of stuff going on in his life"*.

Third, each artist's or actor's *personality* also can be a source of friction. For instance, I4 mentioned that it was hard to collaborate with a band member who had a different work ethic, who *"does not appear on the concert day"* and *"does not come to the practice sessions"*.

Limited Solutions While interviewees brought up problems caused by non-artistic factors, they did not describe many solution approaches for them. In many cases, the solution may have just been to sever the relationship and not work together in the future. In some situations, the required changes were outside the scope of the support relationship, requiring higher-level changes (teams, societal, etc.). For example, I9 mentioned that the fair contract for freelancer visual designers should note that *"the price would be different according to the rounds of feedback"*. I9, however, was less optimistic about it due to the entrenched convention of paying artists little. The only realistic approach that could be done by the artist was to 'give up' on certain things. For example, I12, who works in the game industry, mentioned: *"As for my, like, personal value, I do like drawing stuff from the beginning to the end. But in order to work in the industry, that's something you cannot have, or acquire."*

2.5 Discussion

2.5.1 Informing the Design of CSTs

The analysis identified different aspects of artist support networks: types, support provided, dynamics, and success conditions of relationships. My interest in these features is specifically motivated by how they might inform CST design. In some situations, a CST may be designed to replace, partially or fully, a human actor that was previously part of the support network. In these situations, it is useful to understand the dynamics and function of these existing relationships. In other cases, a novel CST may augment or automate some work the artist currently does. Here, it is valuable to acknowledge the mechanism by which agents come into an artist's support network and what makes those relationships work. Finally, though it may not be the goal of a CST designer to directly map to an existing human actor, there is always the potential that artists will evaluate these technologies through an anthropomorphic lens. Thus, human-human interactions in a support network are highly likely to shape the artist's perceptions and acceptance of a technological tool.

To make this more concrete, I first focus on how findings can inform the design of steerable AI-CSTs. Then, I focus on two CST types: AI-CSTs and collaborative CSTs. The first category of interest is AI-CSTs, or CSTs powered by AI. As these CSTs can potentially automate some roles played by actors, the artist might expect these tools to have interaction patterns similar to those of human actors. The second category, collaborative CSTs, enables the involvement of multiple individuals in the creative process. The tools directly act on the artist support networks themselves. Thus, patterns found in support networks can inform how technologies should be designed to have the highest chance of success when embedded in those networks.

2.5.1.1 Informing Design of Steerable AI-CSTs: Iteration and Multiple Communication Means

From how artists get support, we could draw some aspects that lead to the design approaches for steerable AI-CSTs. At a high level, artists and supporting actors tend to steer each other's behaviors so that what the counterpart does falls within their expectations. Communication of their intents and exchange of idea-wise contributions would be the main means to steer the counterpart's behaviors. Such steerability would also be expected in the use of AI-CSTs, whose behaviors can potentially fall out of the user's expectations as they provide autonomous functions, such as the generation of artifacts. In addition to the general need for steerability, I could also identify two specific lessons for designing steering interactions. First, AI-CSTs would need to facilitate iteration in steering interactions. When people communicate intents and build artifacts out of what they communicated, they tend to work iteratively, having another round of communication and building after seeing the intermediate version of the artifacts. It is because single-shot communication and support are not often enough to satisfy the artist, as their languages and ideas tend to be under-defined. Users of AI-CSTs would experience similar issues with under-defined languages and ideas. For example, for those tools that generate a portion of the artifact, the user might input loose specifications on the artifact to be created if they expect some interesting surprises. Hence, the design of AI-CSTs should consider facilitating iterative steering interactions. Second, AI-CSTs can leverage multiple modalities so that users can effectively communicate their intentions. For example, artists and supporting actors often use multiple communication means, from verbal descriptions to references and sketches. Each of them has its advantages and disadvantages. For example, verbal descriptions can be easy to specify, facilitating the iteration, but can be ambiguous. On the other hand, references and sketches can be effective in conveying concrete meanings, but they can be costly to collect or create. When used together, these communication means would potentially complement each other's limitations and facilitate communication. In AI-CSTs also, users would convey their intentions more clearly by using multiple input modalities. In summary, assuming that users would likely expect dynamics similar to what they have experienced with already-intelligent agents, people, AI-CSTs would need to be designed with considerations on iterations and facilitating multiple input modalities. We expand this discussion on iterative steering and multiple modalities as a design element in Chapter 4.

2.5.1.2 Informing Design of AI-CSTs

Apart from implications for iterative CSTs, the analysis of human-human networks can give additional insights on how artificial agents should be designed to replace or supplement human collaborators. This is critical, as designing AI tools has led to long debates on how automation and control should be balanced in human-AI systems [156, 257, 345, 366]. To help designers resolve these tensions, researchers have proposed various high-level guidelines for developing human-AI systems [13, 306]. Many of these are inspired by experiments in various forms of automation (e.g., [156, 345]). While art-making is one of the domains where AI technologies (e.g., generative algorithms) are expected to add values [44, 66, 128, 139], it is not entirely clear if these guidelines apply in the artistic space. Prior analysis of the 'value of artifacts' in this space depends on 'artistic values' (e.g., authenticity [296], authorship [276], and novelty [272]). This is confirmed by some of my analyses–and suggests that automation and guidelines that ignore these values will be problematic.

A few specific efforts have sought to characterize the intersection of human-AI systems and CSTs. Speculative and theoretical work [77] hypothesized ways in which artists may interact with AI-CSTs (e.g., as colleagues with which we collaborate over interactive 'rounds'). This anthropomorphization is possibly reflective of how artists do, or should, interact with machines. However, this presents only one possible approach. Other efforts studied specific interaction techniques of existing AI-CSTs and found design patterns in controllability [63], and the 'split' of the creative process between human and machine [84, 379]. However, no effort was motivated by user needs or expectations in art-making.

All this suggests that existing literature and guidelines have limitations to inform the design of AI-CSTs. For example, existing AI design guidelines emphasize scoping the AI's capabilities (e.g., 'Set the right expectation' [306] and 'Make clear what the AI systems can do' [13]). This is reasonable, but leaves unstated the implied need that the AI's capabilities should align with what specific user populations would expect from them. My analysis of artist support networks concertizes these expectations. Second, while some guidelines resonate with my findings, they may not be detailed enough to convey specific advice in art-making. For instance, when describing good ideation support, we saw that artists hope for novel ideas. However, a good support partner knows that ideas that are too distant are distracting. The need for this balance can inform AI-CST design as the uncertainty inherent in AI systems can be a double-edged sword: proposing both inspirations or distractions. Existing guidelines do not suggest specific design approaches for these situations. At best they may hint at providing controls to align user expectations to uncertain machine behaviors (e.g., 'Encourage granular feedback' and 'Provide global controls' [13]). In many cases, these control interfaces use terminology and affordances appropriate for AI and GUI design. However, this may be undesirable, as it is an open question at which level of details controls and feedback should be provided for art-making AI tools. Understanding how humans do communicate or expect to communicate their needs to other humans can help in designing appropriate interfaces.

I suggest that my findings can help clarify and refine broad design guidelines. To reflect on what artists expect from AI-CSTs, we can assume that situations in which artists have successfully found human collaborators may be amenable to replacement or augmentation with CSTs. The 'sites' of interaction between artist and actor (i.e., the edges in my graphical model) are potential locations into which a CST can be 'spliced.' That is, these locations have already been indicated as opportunities for an artist to utilize other actors. Furthermore, the articulation of relationship dynamics implies a complexity to certain relationships. That is, some interactions (e.g., subcontracting) may be fairly simple and thus more amenable to CST augmentation. We can hypothesize that where sites of human-human relationships exist and where the dynamics of those relationships are simpler, it may be easier to add a CST. For example, automatic comic flatting tools (the process of creating areas for different colors on line art [433]) could be easily accepted by comic artists, as its interaction with the artists would be a simple subcontract relationship (Figure 2.3a). This adheres to the model where the high-level artistic vision (e.g., selection of colors, specification of colors in each cell) is decided, controlled, and implemented by the artist, and the tool implementations a facet of that vision. In contrast, an AI coloring tool that is broad and imposes its own style would lead to an unexpected configuration (e.g., a Featuring relationship as Figure 2.3b) that deviates from the expected subcontract model. Or perhaps worse, the CST would not fit any known support model.

With a knowledge of artists' expectations on support patterns, we can give more specific design guidelines for AI-CSTs. For example, when artists communicate their intentions, the level of details varies based on relationship types. For instance, they would be less detailed in 'featuring' than in 'subcontract' relationships, as the artist would not want to limit the actor's creativity with concrete directions. Such dynamics might transfer to the artist's expectations on AI-CSTs, with more fine-grained controls for "subcontract"-like AI-CSTs.

I emphasize that findings are based on an analysis of what works within human-human support networks. Thus, we can only speculate on how artists will interact with AI-CSTs—they might interact with actors either similarly or differently. I expect my findings to most likely apply to tasks and domains where the artists already have a strong convention on the creation and support process. In such domains, the artists are more likely to expect that support dynamics of the tool mimic their human counterparts. Deviations from this may be tolerated if the benefit of the tool is high, but we hypothesize some difficulty in incorporating the tool into existing work practice. On the other hand, if AI-CST designers offer completely novel support, with no basis for comparison, the tool may be better accepted.

2.5.1.3 Informing Design of Collaborative CSTs

I briefly offer ways in which our understanding of human support networks can inform the design of collaborative CSTs[3]. CSTs in this space are already designed to fit within, or enhance artist-support networks. These CSTs focus on improving the fluency of social interactions for collective art-making. However, such tools often assume equal power among participants. As described in findings, and as supported by past work, shared or equal power is not often the case in actual human-human collaborations. Given this, tools that consider the asymmetries of real artist support networks are more likely to succeed. CSTs of this type can include pairs of 'artifact creator' and 'feedback giver' [209, 297]; leader and crowd [210, 263]; or highly specific relationships such as dancer and choreographer [374].

Understanding artist support networks can also give us insight and inspiration for the design of collaborative CSTs. For example, consider the artistic frictions found in real networks (Section 2.4.3.1). I found that nuanced frictions are caused by gaps in: *trust in understanding styles and values*, *trust in skills and quality*, *communication*, and *concentrating power*. In this context, collaborative CSTs can address the gaps by creating (or being) boundary objects [234, 380, 381], thus serving as a means of translation between artists and actors. For example, learning about another person's artistic styles and values takes a significant amount of time and interaction (e.g., when developing a subcontracting relationship). CSTs can accelerate this process. For example, we might imagine a tool that summarizes an artist's style or work process by mining their work or log data and then presenting the information to the other party. Analysis of this type can also help in situations where collaborating artists and actors might not be aware of all knowledge and skills required for fluent collaboration. We could imagine a CST that supports a vocalist and composer's collaboration by analyzing and describing the performer's vocal range. In situations where the artistic languages of the collaborators are different, CSTs can enhance communications by building spaces for shared 'models' to represent what each artist means when they say something. For example, we might imagine a 'surface' where artists can collect instances of what they mean when

[3]Note that while this category is broad and can extend beyond art (e.g., crowd ideation [369]), I mainly focus on variants for collaborative art-making

Our work.			Nakakoji. (2006)	Frich et al. (2019)		Mamykina et al. (2021)
Artifact-influencing Support	Implem-entation	Expertise	Running Shoes	Implementation	Iteration	Construction
		Labor				
	Ideation	Feedback		Critique		Evaluation
		Inspiration	Skis	Ideation		Creative Concept
Artist-influencing Support	Manage			Project Management		
	Teach		Dumbbell			

Figure 2.4: Comparison of taxonomy on types of support.

describing 'dark' or 'light' concepts. Finally, as collaborative friction can come from implicit power structures between users, CSTs can facilitate the awareness by making them more explicit. For example, CSTs can require participating artists to specify their roles and build more explicit 'contracts' and workflows. The design of the tool in Chapter 5 is largely based on lessons on collaborative CSTs: the tool supports the human-human collaboration of an asymmetric relationship while facilitating the use of shared boundary objects.

2.5.2 Comparison to Previous Models on Artistic Support

My analysis of artistic support resonates with taxonomies and findings from previous work. While my novel contribution is mainly on identifying dynamics in different types of relationships, I also extend the taxonomy of support[4] and frictions. I have attempted to achieve more comprehensiveness, with wider coverage and finer granularity.

By explicitly articulating the support dimension in the artist support network (Figure 2.4), I have identified aspects that may not only apply to human supports but to tools as well. My taxonomy categorizes types of support into two higher-level groups, *artifact-influencing support* and *artist-influencing support*, which tell us different mechanisms of getting support. On the other hand, previous work lacked this distinction and represented support as a 'flat' structure. Previous taxonomies also did not cover the full range of *artist-influencing support*, while I identified both *managing* and *teaching*. This may be due to the different focus: I tried to consider all supports that

[4]I note that my idea of "support" is sometimes expressed as "roles" or "activities" in some previous work.

Our work.		Mamykina et al. (2002)	Luther et al. (2010)	Luther et al. (2013)	Settles et al. (2013)
Trust in Understanding Styles & Values		Common Artistic Intentions & Visions			Stylistic Similarity Does Not Directly Affect Success.
Trust in Skills & Quality		Share Knowledge	Reputation & Experience	Problem solving	
Communication		Share Common Language	Communication & Dedication	Informing & Monitoring	Communication
		Engage in Extensive Discussion & What if Sessions			
Concentrating Power	Selectively accepting other ideas				
	Dictating		Planning & Structure	Planning & Clarifying roles/objectives	Balance Effort
	Relinquishing power				

Figure 2.5: Comparison of taxonomy of approaches for successful support relationship.

artists get from other actors that are intended to help. With this scope, I could comprehensively identify types of support, even those that do not directly contribute to artifacts. Similarly, while the previous work did not consider commonalities between critique and inspiration, I grouped them under the higher class of *ideation*.

For solutions to friction (Figure 2.5), I identified a large set of approaches. In interviews, I observed solutions that were identified in previous taxonomies. Moreover, I also found an additional category, *selectively accepting other ideas* within the solution of *concentrating power*, which was missing in other taxonomies. Furthermore, my work could find to which relationship types each solution would work well. This difference would be because I investigated diverse art-making settings, while prior work focused more on a specific one. Another interesting aspect to note is that Settle et al. [357] found that stylistic similarity does not necessarily lead to success in support. On the other hand, my closest category, *trust in understanding styles and values*, indicates that understanding and trusting different actors' styles is more important than having the same style in artistic production.

I also relate my findings to broad collaboration research within HCI. In the literature on shared leadership [449, 450], researchers identified four types of leadership behaviors, from giving positive or negative feedback to directing and promoting social engagement. Some are relevant to solution approaches from my findings. For example, being directive would be relevant to *having communication* and *concentrating power*. However, other aspects of leadership behaviors would be more relevant to non-artistic frictions discussed in Section 2.4.3.2. For example, artists giving adequate feedback and engaging socially with actors would be solutions for non-artistic frictions.

2.5.3 Limitations and Future Work

I purposefully studied support patterns in a wide range of art creation domains. However, this approach limits our ability to learn detailed support patterns in a specific art domain. Thus, one future direction can be to extend my analysis further in one specific domain. This will enable us to find specific sub-categories that may be of interest and importance within that domain.

Again, due to focus, there may be rarer types of relationships that my taxonomies might not fully explain. For example, take the case of a writer finishing a book for a deceased author. In this case, the living writer, like a *subcontractor*, is expected to follow the style and outline of the deceased. At the same time, because there is no absolute guidance, like a *featuring* writer, the writer is also expected to make some creative decisions that shape the work. Though rare, this type of relationship is rather unique and does not as cleanly fall into one of my high-level categories.

In my work, I focused on supports that are intended—either by human actors who are willing to help or by tools that are designed with intended support functions. However, there may be other types of supports that are not intended, but still influence artists. Receiving inspiration from an artist who died 200 years ago is one example of such a case. As these types of support might show different dynamics, investigating them is worth studying in future work.

As mentioned in Section 2.5.1.2, I presented a possible use of my framework in the design of CSTs. A more formal study of this approach is an important next step. A more formal mapping of the existing CST design to my framework would also help identify situations where mapping can be made directly and where it can not. For example, many CSTs are much more focused on their functions than human actors. Further research would also help us examine whether and how an artist's expectations for AI-CSTs relate to their analogous human roles. This type of analysis would allow us to extend my model to account not only for human actors but CSTs as well.

2.6 Conclusions

In this work, I investigated various types of relationships within the artist's support network. I conducted interviews with fourteen artists from a wide range of domains to understand common types of relationships, their dynamics, the general roles filled by support actors, and the reasons and problems (the frictions) inherent in these relationships. Additionally, I identified patterns of how these factors relate to each other in different support relationships. I motivate this investigation with the design of CSTs, which automate and augment the artist's support network. I believe that perceptions and expectations of effective human support are likely to influence the way artists view the design of CSTs. Based on my interviews, I first identify the artistic need behind considering iteration and multiple input modalities as core design elements for steerability in AI-CSTs. In addition to that, I offer other implications for designs of AI-CSTs and CSTs for collective users. In Chapter 4, I connect back to findings and discussions from this chapter and introduce design requirements and lessons for steerable AI-CSTs.

CHAPTER 3

The Intersection of Users, Roles, Interactions, and Technologies in Creativity Support Tools

3.1 Introduction

Technological innovation has always led to changes in art-making. Computer-based tools, in particular, have enabled extended expressions, efficiency, and skills [118]. For instance, Adobe's Photoshop provides a set of tools ranging from assorted brushes to context-aware selection tools. Many features mimic existing physical tools and infrastructure available to artists but some afford new capabilities for which there is no equivalent. As *technologies* evolve to support new forms of *interactions* and information processing, these capabilities have been integrated into CSTs [119, 364, 365]. For example, many traditional CSTs lack 'agency.' However, new advances in AI and ML have enabled CSTs to become more autonomous, allowing them to do tasks on behalf of the user. This has naturally led to new forms of interaction. An *intelligent* paintbrush–one that does not simply put paint on the material but renders objects in the appropriate style with simply an outline–does not operate like a standard paintbrush (digital or otherwise). Combinations of *technologies* and *interactions*, can expand the capacity of CSTs to serve a wider set of *roles*. For example, with advanced generative algorithms such as GAN [139, 308] or style transfer algorithms [128, 360], CSTs can serve the *role* of creating a portion of the artifact on behalf of the users. The combined changes in *technologies*, *interactions*, and *roles* has also led to an expanded range of supported *users* [119, 120]. For example, CSTs that autonomously generate artifacts lower the creation hurdle for users, allowing novices to create artifacts with minimum skills.

This chapter builds on past reviews of CSTs [119, 120, 335]. I specifically seek to provide a framework for understanding the intersection of *technologies*, *interactions*, *roles*, and *users* in shaping *art-making* CSTs. Investigation in this chapter is ultimately to have a better understanding of the designs of AI-CSTs. By taking an integrative approach that considers various aspects of CSTs, I aim to map the design space for CSTs. Mapping this space has a number of benefits. For example, we can better identify what are effective design approaches for different types of

CSTs. I can also learn what areas are under-explored, difficult, or not yet technologically viable. These gaps hint at future research directions: from those that require more attention; to emerging opportunities and challenges; to those that would be more plausible with technological advances.

In this chapter, I conducted a literature review of 111 publications that introduced novel art-making CSTs. From the analyzed papers, I identified various facets to model CSTs. These include a resource-based model for understanding CSTs and their placement in the creation process. Additionally, I identify dimensions of interaction approaches to better understand the models of interactions between the human and tools. My analysis revealed a broad range of technologies by which CSTs are implemented. Finally, I connect these taxonomies to user types and usage scenarios.

Through the analysis, my work contributes a comprehensive understanding of the design space of art-making CSTs while giving hints at future research directions. I identify patterns that are common within CSTs created by the HCI community. These patterns can inform future CST designers and allow us to identify under-explored CST types. Specifically, I could identify patterns with CSTs that are powered by machine learning algorithms. I could find that 1) AI frequently powered tools that provide idea-wise support, 2) AI-powered CSTs tend to be unpredictable to users, and 3) they frequently add implementation directly to the user's artifact. To address the potential issue of unpredictable implementations, previous CST designers adopted control approaches in these CSTs. Moreover, from the patterns of how tools turned complex steering controls into easy and efficient interactions, I could learn that such design patterns would also benefit the design of AI-CSTs that consider many aspects to control. I conclude this chapter with how my findings relate to the need and design lessons for steerable AI-CSTs. This chapter is based on the paper "The Intersection of Users, Roles, Interactions, and Technologies in Creativity Support Tools," published at DIS2021 [63].

3.2 Background and Related Work

Tools often play an important role in [25] and are themselves shaped by advancing technologies [29]. Computing technologies have enabled a broader class of CSTs, and their development and study have become a fixture in the HCI community [364, 365]. The evolution of these tools in the research community has been extensively tracked [119, 120]. This work has provided a lens to study the trajectory of CST research and associated focus areas. I build on this work not only to map work within the research community but also to identify a design space for art-making CSTs.

While different efforts to taxonomize CSTs with roles have been introduced in Section 1.1.1, an alternative structure for the study of CSTs focuses on evaluation. For example, Garfield [127] proposed that CSTs can be evaluated with products' quality, such as novelty or appropriateness. Similarly, Carroll et al. [50] and Cherry et al. [59] introduced Creativity Support Index (CSI), evaluating CSTs in six criteria: exploration, collaboration, engagement, effort, tool transparency, and expressiveness. Others have taken a more critical view of CST evaluation, identifying a lack of clarity in evaluation goals, theoretical grounding, and expert participants [335]. Models of evaluation are of obvious importance. However, this only represents a facet of determining the effectiveness and appropriateness of CST development.

It is worth emphasizing that CST research is extremely broad. Not all CSTs focus on art-making (e.g., those for inventions [135]). Art-making is a relatively unique and complex space for human-machine interaction. Values such as ownership, authenticity [92, 296], and the end-users intrinsic motivations (e.g., what is enjoyable?) [199] will shape the effectiveness and acceptability of new tools. The introduction of CSTs with cutting-edge technologies in the academic community—including those of AI—will invariably turn into commercial products, making this space even more complex. Thus, I am interested in understanding how technologies and interactions have shifted CSTs for art-making. The notion that technologies impact art-making is by no means a new one. Technological innovation in the arts often has increased production efficiency and enabled diverse expressions [29].

Computing technologies have brought multifaceted changes to art-making CSTs. One change is in how CSTs interact with the users. For example, AI or ML has shifted CSTs to be more autonomous and unpredictable [78, 79, 277]. These advanced technologies and diversified interactions have also driven CSTs into a broader set of roles. For example, CSTs can adopt advanced recognition algorithms to automatically provide critiques [326, 367]. Generative algorithms allow CSTs to build artifacts on behalf of users [69, 147, 256, 277, 301, 415]. Art-making CST innovations have enabled new forms of interaction for more diverse users, from novices [211, 256] and experts [130, 217] to those with disabilities [294]. Taken together, these innovations represent an opportunity for synthesis.

In this chapter, I build upon past efforts in mapping CSTs. However, I have specifically targeted art-making CSTs for analysis. Because of this focus, I identify unique categories and features. This chapter also seeks to better contrast different taxonomic categories for CSTs (i.e., roles, interactions, technologies, and users).

3.3 Literature Review Method

For the analysis, I implemented a sampling strategy and a coding process to identify and analyze papers for art-making CSTs.

Table 3.1: Reviewed publications according to sampling approaches.

Source	Publications
From Frich et al. [119]	[28, 57, 67, 69, 78, 79, 80, 124, 126, 143, 150, 164, 172, 193, 200, 201, 204, 211, 273, 291, 304, 319, 398, 399, 400, 415, 429, 431, 438, 446, 448]
Newly sampled.	[7, 17, 19, 55, 61, 87, 115, 122, 130, 131, 141, 151, 160, 169, 171, 184, 194, 196, 197, 217, 218, 219, 220, 224, 225, 237, 245, 248, 253, 254, 294, 311, 317, 320, 355, 356, 361, 367, 368, 383, 393, 397, 421, 423, 426, 428, 432, 441, 444]
Exploratorily found.	[1, 21, 31, 51, 94, 95, 113, 116, 117, 121, 147, 155, 165, 186, 189, 230, 236, 243, 247, 256, 264, 277, 281, 301, 315, 326, 362, 388, 392, 422, 435]

3.3.1 Sampling

I first set the criteria that decide which CSTs to sample. While I focus on art-making CSTs, notions of ‘art’ and ‘art-making’ are the subjects of significant academic and philosophical debate [76]. I take a fairly broad definition, scoping “art-making” CSTs as *systems or tools used as part of the creative process that result in an artifact with aesthetic qualities*. In my definition, artifacts can take various forms. In music, gaming, creative writing, and film-making, artifacts can be demos and footage that are stored in digital formats. Artifacts can also be form-changing sculptures, sketches, and embroidery in domains such as sculpting, painting, and fiber-based art, respectively. In my model, artifacts can serve as components or instructions for larger creative ‘result.’ For example, a script (for animation) or dance movement annotations (for a dance piece) are also artifacts. While this definition is broad, there are CSTs that we exclude. Most often, these tools are unlikely to lead to the creation of an artistic product. For example, I do not consider tools for business decision making, crowd tasks (e.g., ‘design a way to remember a person’s name’ [54]), or artifacts that do not consider aesthetics or artistic values (e.g., practical inventions [135]). Additionally, I limited the scope of this study to single-user interactions around art-making. Hence, I excluded CSTs that *only* supported the collaboration or communication between artists, but did not directly support art-making (e.g., a system that redistributes responsibility and leadership in web-based art-making collaborations [263]).

With my criteria, I focused on surveying CSTs from research (specifically HCI). With this approach, I can investigate novel art-making CSTs that have not yet made their way into commercial tools. With novel academic tools, I can gain a sense of the reactions (and possibilities) inherent to technologies that are not yet commercially available. A second, more practical reason is that research papers are much more explicit in explaining the intended roles, designs, technologies, and users. With the descriptions in the paper, I could also decide whether the tool is within my criteria.

To begin the sampling, I leveraged the literature review of CSTs by Frich et al. [119]. I identified those CST papers that fell within the inclusion criteria for art creation. This initial pass yielded 31 papers.

To expand this set, I identified post-2018 publications (the end year of Frich et al's survey). I targeted papers published between September of 2018 and October of 2020. To sample these, I took an approach similar to Frich et al. I used the author keywords 'creativity support tool' or 'creativity' to search and sample papers from the ACM Digital Library.[1] Unlike Frich et al., I did not filter papers with download or citation count. It is because bibliometric measures tend to be relatively small for all newer papers. Among this set, I identified 49 new papers that follow the criteria.

I also sought to include relevant papers that did not have author keywords of "creativity" or "creativity support tools." This was done by exploring publications that cite or are cited by the sampled papers. I also searched through proceedings of recent HCI conferences. These yielded additional 31 papers. In total, I considered 111 publications from 2009 to 2020 (Table 3.1).

3.3.2 Coding Process

With a co-author, I analyzed sampled papers iteratively. I focused on 1) the purposed roles of the tool, 2) the interaction patterns between users and tools, 3) technologies used, and 4) intended user population. I targeted these factors based on previous work (Section 3.2), as they either drive changes in CSTs (technologies) or are impacted by those changes (roles, interactions, and users). The analysis goal was to find patterns in, and between, these factors.

The iterative approach involved multiple joint sessions of discussion and analysis with the co-author. Each session added up to 10 papers. Papers for each session were randomly sampled from the papers that had not been reviewed yet. Before each session, I and the co-author independently read, summarized, and coded sampled publications. Codes were developed over the course of the sessions. When appropriate, I also leveraged codes from existing work as a starting point.

If a new paper fit under an old code, this was used. Otherwise, I and the co-author inductively generated codes as we went through the data. Section 3.4 describes which codes were based on previous work and which were inductively generated. In each session, I and the co-author reviewed each other's summaries and codes. For generated codes, we tried to integrate differing codes into a single scheme. If necessary, codes were revised, removed, added, merged, or split. The updated coding scheme was used for the next rounds of discussions. Moreover, with the update of the coding scheme, I and the co-author reviewed past codes and updated them with the new scheme. After analyzing the whole paper set, we identified additional higher-level structures.

[1]https://dl.acm.org/

3.4 Codes

Through coding, I structured taxonomies of roles, interaction approaches, technologies, and users of CSTs. Some codes were grounded in previous work, while some were inductively found during the coding process. In this section, I introduce details of each code in taxonomies. All codes are summarized in Table 3.2.

3.4.1 Roles of CSTs

I identified roles with two different taxonomies. First, ***resource roles*** indicate which type of resources, or benefits, each CST offers. These further divide into two types based on whether the resource was an 'idea' or something more tangible and skill-based, like 'labor' or 'expertise'. In contrast, ***process roles*** indicate in which part(s) of the art-making process the CST is intended to work. At a high-level, *process roles* include ***aiding ideation***, ***aiding implementation***, and ***aiding evaluation***. These are grounded on the creative process phases identified by Amabile [8, 9]. My coding approach is similar to those of a CST's roles by Frich et al. [119] and the Library of Mixed-Initiative Creative Interfaces [379]. Based on previous work and the coding process, I further identified which more specific *process roles* exist under each high-level role.

3.4.1.1 Resource roles

I identify two types of *resource roles*: (1) those that help with ***implementations***, and tend to support artists with expertise or labor efficiency; and (2) those that help with ***ideas***, and target artistic vision or ideas. A simple example within the *implementation* category is a tool that helps if an artist cannot implement an artifact due to the lack of expertise (e.g., they may not be able to sculpt well) or time (e.g., a complex pointillist piece). In contrast, *idea*-focused CSTs can offer an inspiring suggestion or create part of the artifact that the artist could not have brought up by herself. In my analysis, I did not find examples of CSTs that focused on both roles. This is not to say that a complex CST could not do both. For example, a full platform for image editing such as Photoshop might arguably offer both in different parts of the system. However, such scale did not exist in the academic examples I studied and, arguably, one might divide a monolithic CST like Photoshop into smaller features/components.

3.4.1.2 Process roles

My analysis identified a more complex set of *process roles*. Additionally, I found many CSTs with multiple process role codes.

Aiding ideation The first high-level *process role* a CST might have is supporting the user's ideation process. Artists or designers tend to seek novel and inspirational ideas before or during creation. A specific instance of this role is ***idea generation***. For example, Karimi et al. [196] used sketch generation algorithms to inspire designers when they are doing a visual design task. A second code, ***curation***, plays a similar role but focuses on suggesting from existing information or artifacts. For example, Koch et al. [218] designed an intelligent mood board that curates contextually appropriate inspiration images for designers.

Aiding implementation The second high-level process role is helping with the implementation of artifacts. Here, a CST augments or automates certain functions. One specific sub-category is ***execution assistance***. For example, Dynamic Brushes [186] help artists create procedural visual arts without programming knowledge. This is achieved by providing a custom interface that lowers required expertise. A second variant, ***producing*** was assigned to situations where artists let CSTs conduct most of the implementation tasks. In this role, the artists can allow the CSTs to make most of the creative or implementation decisions. Users who lack the expertise or labor to implement artifacts by themselves benefit from this category of CST. For example, Frid et al. [121] designed a music-producing system for video creators who don't necessarily know anything about composition. The ***understanding*** code was used for CSTs that help the user understand the current state of their artifact. This role is helpful when the implementation of creations can be complex. For instance, Progression Maps [51] help interactive narrative designers understand complex narrative structures through a visualization. This role is different from the other two specific roles in *aiding implementation* as it does not directly support the artist's implementation of the artifact. Rather, *understanding* helps with the sensemaking required for implementation.

Aiding evaluation The third process role was in helping with the evaluation of created artifacts. Within this role, I identified one specific code, ***critique***. Here, the CST critiques or gives feedback intended to guide improvements to the artifact. For example, VoiceAssist [355] gives feedback on voice recordings, so that users can make improvements on room acoustics and background noise.

3.4.1.3 Complementarity of Resource and Process Roles

There are situations where process and resource roles are strongly connected. For example, giving someone an 'idea' often happens within 'ideation' (a specific process). While this may be more common, one can imagine situations where ideas are provided in other points in the artist's creative workflow (i.e., process). For example, when evaluating an artifact, the system can also provide ideas for improvements. Because of this, I treat resource roles and process roles as complementary.

By splitting role types, I can distinguish between the *benefits* offered by the CST and *where/how* they are offered within the creative workflow.

3.4.2 Interaction Approaches Used by CSTs

My analysis identified the interaction approaches used within CSTs. I found that a single tool can have multiple interaction behaviors corresponding to multiple functions. While the traditional types of interactions (e.g., mouse, voice, touch, direct manipulation, etc.) are part of this analysis, I am more concerned with the properties and intents of the interaction relative to the creative process.

3.4.2.1 Input Directness

I categorize input directness in relation to whether a CST is receiving direct inputs or not. Here, ***direct*** inputs are closely relatable to the artifact or the change that is going to be made in the artifact. A simple example of a direct input is brush strokes on a digital canvas. Clearly, these directly relate to what is drawn on the canvas. It is important, however, to note that I distinguish between the idea of 'direct manipulation' and 'direct input.' A more subtle example of direct input is when the input becomes part of the artifact's final 'form.' For example, the recorded audio of a voice-actor can be recorded when producing a character animation. That voice is used to drive the expression in the character but is also a final part of the animation (as in Adobe's Character Animator and TakeToons [388]). I also consider the observation of the "current state" of the artifact as *direct* input. For example, a CST can take the current representation of the artifact as input and provide a critique of that work. While the art is not being *modified* by the CST, the input is nonetheless direct, as the input is the artifact itself. ***Indirect*** inputs are those that are more separated from the artifact. One example is natural language queries given by the user. These queries would be used to request various functions to the tool (e.g., searching or generating artifacts), but queries themselves are not artifacts. Another type of *indirect* input is a manipulation of parameters, like those for cameras, such as exposure levels. It is more of partial information about how the artifact should be, but not the representation of the artifact.

3.4.2.2 Predictability of Impact

The second property of interaction is *predictability* of the CST when it is used. How well can the user model and anticipate what the tool will do? A ***predictable*** CST is one in which the CST behaves exactly according to the user's specifications or anticipation. An example of the former is a brush in virtual canvas, where users are certain that the lines will be created following their strokes. An example of the latter would be a fabrication tool that receives a blueprint from the user. While the blueprint may not be a complete specification (e.g., it may not contain scaffolding instructions), the user is nonetheless certain about the tool's behavior and what the final output will look like.

CSTs that are ***unpredictable*** are those that produce output that is difficult for the end-user to model. These tools are clearly not 'random'—the overall function is understood. I can take the example of the 'art-critic' CST that is constantly providing feedback on the art. The end-user is aware that critiques are being produced but can't accurately model what they will be. *Unpredictable* tools rarely require users to give very specific information on how the tool should behave. In fact, it is this ambiguity that makes them unpredictable. While tools can be *unpredictable* due to errors, we coded CSTs according to their intended behaviors. I also recognize that CSTs can be both predictable or unpredictable. For example, an unpredictable CST can become predictable given enough experience.

3.4.2.3 Output-Implementing/Influencing

The final aspect of the interaction codes is on the *output* of the CSTs. I identified differences in CSTs that support the artist by ***implementing*** the whole or a part of the artifact or by ***influencing*** the artist. I coded tools to be *implementing* if they directly create or generate a part of an artifact. An example is a generative algorithm that creates the visual design of products [326]. I also coded tools as *implementing* if they simulate the final artifact or some part of it. SimuLearn [435] is one example of this type of CST. The tool generates a simulation of how the 3D fabrication would change with the application of heat. I coded as *influencing* those CSTs that impact the artist, not the artifact. This includes feedback, critique, scaffolds, or analysis. The artist is intended to *react* to this information in modifying their behavior, and thereby, the artwork. In some cases, the CST's output is both *implementing* and *influencing*. This can happen in the cases of mixed-initiative systems. For example, in a tool that outputs inspirational metaphors for a word [131], the metaphor can be directly used in the user's writing or can influence artists to get inspiration and draw more ideas.

3.4.3 Technologies for CSTs

When considering the technologies used in CSTs, I focused on the aspects that provided core functionality or interaction support. I inductively identified six types among the sampled papers.

3.4.3.1 Learning algorithms

CSTs based on ***learning algorithms*** were those that were trained on data. They include many ML algorithms, ranging from Hidden Markov Model, neural networks, and Generative Adversarial Network [139, 308]. One use of these algorithms is to recognize and understand artifacts or user inputs. For example, Shtern et al. [367] used ML recognition algorithms to inspect the quality of music mastering. *Learning algorithms* were also used for generating artifacts. For example, McCormack et al. [277] designed a machine improviser that generates music that fits with what a human musician is playing. *Learning algorithms* were also used to learn users while they are using CSTs, so that tools can give adaptive support. For example, Drawing Apprentice [79] learns how to co-create with the user from user interactions.

3.4.3.2 Non-learning algorithm

CSTs that were not data-driven were classified as ***non-learning algorithms***. This type included hand-tuned, rule-based algorithms, or optimization algorithms. CSTs in this category often leveraged these algorithms for artifact exploration or search given some constraints. For example, Scout [392] suggests designs with a constraint resolver and a ranking function. Algorithms used in Scout were designed by researchers to assure the quality of the suggested designs. *Non-learning algorithms* are also used to manipulate the artifact in a controlled way. For example, DataQuilt [444] used GrabCut [344] and Canny edge detection [49] to extract specific parts of images. I coded *non-learning algorithms* separately from *learning algorithms* as they are different in how they are designed. This difference would possibly impact how CSTs with these algorithms would interact with the users. For example, *non-learning algorithms* are more often designed to perform deterministically according to the tool designer's intention. However, *learning algorithms* are often trained on data—both small and large—that leads to uncertainty in how the tool behaves.

3.4.3.3 Software UI

CSTs that were principally centered around ***software UIs*** often involved designs to improve user control of the CST. For example, Demystified Dynamic Brushes [237] helped artists understand the dynamic visual arts by showing relevant numerical parameters. These also helped the editing of dynamic arts by allowing easier manipulation within the UI.

3.4.3.4 Sensors

Sensors have been used in CSTs to expand the modality of the expressions. Their usage ranged from photo-sensing to audio-, depth- and gyro-sensing. For example, MoBoogie [150] allows users to create musical expressions with dancing moves, by sensing them with an accelerometer.

3.4.3.5 Fabricators

Some CSTs use new ***fabricators*** or materials (or leverage existing ones). For instance, ExpandFab [194] introduces a fabrication process of expanding objects using foam materials.

3.4.3.6 Robots

Though rare in the samples, some CSTs had mechanical or ***robotic*** infrastructure. This enabled the CSTs to interact in physical spaces. For example, Robovie [193] is a physical robot designed to give inspiring prompts on garden designs.

3.4.4 Users of CSTs

When classifying CSTs, I also focused on who they were targeted for. I built on the taxonomy of Frich et al. [119] for expertise and augmented this with codes for the populations that were the intended audience of the CST (i.e., general use or specific populations such as children or end-users with certain disabilities).

3.4.4.1 Expertise/Availability of Description

The first set of user codes included four categories corresponding to expertise: novice, expert, all/both, or unspecified. CSTs for ***novices*** enabled creations that are not possible with the user's expertise. For example, ChordRipple [172] helps *novice* composers try radical chords with recommendations. Some CSTs are designed specifically for ***expert*** users. For instance, VoiceCuts [217] allows *experts* to interact with creative applications through vocal commands. For *experts*, as they have their ways of creating, it would be important to design a tool that embeds well into their practices. CSTs can also be designed to support ***both*** types of users, regardless of their expertise. For example, Joinery [448] supports both *novices* and *experts* in the fabrication of laser-cut assemblies. For *novices*, it democratizes the creation, and for *experts*, it provides rapid prototyping ability. Lastly, some CSTs did ***not specify*** target users. For instance, BodyAvatar [446] focuses more on describing the novel tool, without explicitly (or implicitly) indicating for whom the tool is designed.

3.4.4.2 Specific and General Populations

Finally, I considered whether the CSTs were focused on specific or general populations. CSTs for ***specific populations*** often had narrow use-cases in mind (e.g., children). For example, Zarei et al. [441] designed a tool that helps children create storytelling with embodied avatars. Another set of *specific population* CSTs are those built for end-users with some disability. For example, CreaTable [294] supports people with aphasia to create content with tangible interactions. Those CSTs that were either explicitly or implicitly for more ***general populations*** were labeled as such. One commonality in *specific populations* that distinguishes them from *general populations* is that

Table 3.2: Definitions, example CSTs (e.g.), counts (#), and percentages (%) of codes in each taxonomy. Resource roles are types of supportive resources provided by CSTs, and process roles are about which part of the creation process is supported. Directness is an interaction dimension of whether the user input is close to the artifact or not, while predictability is about whether the tool behavior is predictable or not. Output is about how the tool is contributing to the creation of artifacts. Technologies and users indicate which technologies are used, and who are the intended user population, respectively.

	Taxonomies	Codes	Definition	e.g.	#	%
Roles	Resource roles	Vision	The tool supports the user with artistic vision and ideas.	[131]	**63**	**56.8**
		Implementation	The tool supports the user with expertise or labor efficiency.	[438]	48	43.2
	Process roles	Idea Generation	The tool suggests novel information or artifacts with computational generation.	[196]	43	38.7
		Curation	The tool suggests novel information or artifacts from existing sources.	[218]	10	9.0
		Execution Assistance	The tool augments the user's implementation actions.	[186]	**81**	**73.0**
		Producing	The tool automates implementation on behalf of the user.	[121]	14	12.6
		Understanding	The tool helps users understand the current state of creation.	[51]	19	17.1
		Critique	The tool helps users evaluate the created artifact.	[355]	10	9.0
Interactions	Directness	Direct	The user input is close to the artifact.	[388]	**89**	**80.2**
		Indirect	The user input is distant from the artifact.	[423]	82	73.9
		No Input	The user does not make an input to the tool.	[193]	2	1.8
	Predictability	Predictable	The user can predict which output will come out.	[247]	60	54.1
		Unpredictable	The user cannot predict which output will come out.	[61]	**76**	**68.5**
	Output	Implementing	The tool implements the whole or a part of the artifact.	[301]	**96**	**86.5**
		Influencing	The tool influences the user.	[355]	39	35.1
Technologies	Technologies	Learning Algorithm	Algorithms trained on data (e.g., ML algorithms).	[277]	**45**	**40.5**
		Non-learning Algorithm	Algorithms not trained on data (e.g., rule-based algorithms and optimization).	[444]	30	27.0
		Software UI	Software UI that gives easier use and control (e.g., visual programming).	[237]	25	22.5
		Sensor	Sensors that expand the modality (e.g., depth sensors).	[150]	23	20.7
		Fabricator	Fabricators or materials, or new ways of using them (e.g., XY-plotter or thermochromic ink).	[194]	11	9.9
		Robot	Robots in the physical space (e.g., robots that draw sketches).	[193]	3	2.7
Users	Users-Expertise	Novice	Users who are not fully trained in the art-making domain.	[172]	31	27.9
		Expert	Users who are enough trained in the art-making domain.	[217]	30	27.0
		Both	Tools support both experts and novices.	[448]	17	15.3
		Not Specified	Tools not specifying which user group is supported.	[446]	**33**	**29.7**
	Users-Specific	General Populations	Users other than specific populations.	[95]	**105**	**94.6**
		Specific Populations	Specific targeted users, such as children or users with a disability.	[294]	6	5.4

they might require special accommodations—either due to not-yet developed motor or cognitive abilities or disability.

3.5 Coding and Analysis Results

Not all codes are equally likely among CSTs. Below, I provide the distribution of codes in each taxonomy. For taxonomies that allow multiple codes for a tool (process roles, interaction approaches, and technologies), I present co-occurrence statistics. Finally, I return to the main motivating question in how roles, users, technologies, and interactions intersect in the design space of CSTs. I include coding results in the supplementary material.

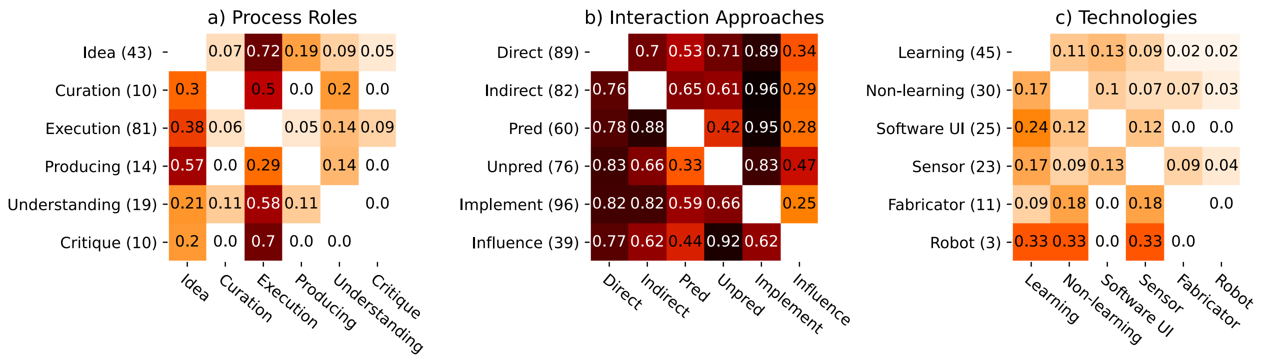


Figure 3.1: Co-occurrences of codes within each taxonomy. The number in each box indicates the ratio of tools that have the element in the column among all tools with the element in the row. On the y-axis, numbers in parenthesis indicate the number of CSTs with the code. In a), 'Idea' and 'Execution' stand for idea generation and execution assistance, respectively. In b), 'Pred' and 'Unpred' stand for predictable and unpredictable, respectively.

3.5.1 Distribution for Each Taxonomy

Our analysis on the distribution of codes for each taxonomy is presented in Table 3.2. For *resource roles*, I found that there were slightly more *idea*-offering tools (56.8%) than *implementation*-offering ones (43.2%). For *process roles*, I found that the percentage of tools for serving *execution assistance* was the highest (73.0%), followed by *idea generation* (38.7%). The rest of the *process roles* had lower percentages, in the order of *understanding* (17.1%), *producing* (12.6%), *critique* (9.0%), and *curation* (9.0%).

For interaction approaches, there were slightly more tools that allow *direct* inputs (80.2%) than *indirect* ones (73.9%). Only two tools did not receive user inputs (1.8%) [7, 193]. With *predictability*, there are more *unpredictable* tools (68.5%) than *predictable* ones (54.1%). With *output* categories, there are over twice more *implementing* tools (86.5%) than *influencing* ones (35.1%).

For technologies, *learning algorithms* were most used (40.5%) whereas *fabricators* (9.9%) and *robots* (2.7%) were least common.

For users, with expertise and the availability of user description, 29.7% of tools did *not specify* user group, which was the highest among user groups. Tools for *novices* (27.9%) and *experts* (27.0%) were slightly less common. Fewer supported both experts and novices (*both*, 15.3%). Only six were for *specific populations* (5.4%).

3.5.2 Within-Taxonomy Analysis

Figure 3.1 summarizes the co-occurrence of codes within the CSTs.

3.5.2.1 Within Process Roles

In *process roles*, *execution assistance* co-occurred most frequently with other *process roles* ($\geq$ 50%, 3rd column of Figure 3.1a) except for *producing*. *Idea generation* also showed relatively high co-occurrences with other process roles ($\geq$ 20%, 1st column of Figure 3.1a). However, *producing* was an exception to this pattern, as it co-occurred more with *idea generation* (57%) than with *execution assistance* (29%). This different result in *producing* is likely due to the incompatibility of *producing* and *execution assistance*: tools with *execution assistance* tend to maintain user control while *producing* tools create on behalf of the user. For cases where they co-occur, separate functions supported each role. Additionally, the high co-occurrence of *idea generation* in *producing* tools would be because *producing* often requires creative decisions.

3.5.2.2 Within Interaction Approaches

I analyzed the co-occurrence in three dimensions of interaction approaches (Figure 3.1b). Within *directness*, more than half of tools were *direct* and *indirect* at the same time (55.9% of all tools). Compared to *directness*, the co-occurrence was relatively low for the dimension of *predictability* (22.5% of all tools). This indicates that *predictability* more clearly characterizes each tool compared to *directness*. Within the *output* dimension, while 21.6% of tools both support implementing and influencing, the rate of *influencing* tools co-occurring with *implementing* tools (62%) was higher than the co-occurrence of the other way around (25%). It is due to the imbalance of frequency within the *output* dimension. When one tool has both codes in one interaction dimension, it was often because the tool has multiple functionalities. For example, COCOCO [256] has an *unpredictable* function of generating a part of the music and a *predictable* function of recording the user's midi input. I also observed that *implementing* code co-occurred with all *directness* and *predictability* codes frequently (5th column of Figure 3.1b). Similar to results within the *output* dimension, it would also be due to the high number of *implementation* tools. Another notable pattern was that *influencing* tools more co-occurred with *unpredictability* (92%) than *predictability* (44%). This result would be because many tools *influenced* artists with unexpected information or artifacts, such as critiques or inspirations. While it was rare for *influencing* tools only to be *predictable*, there were cases that the tool does *predictable* management for the user. For example, VoiceCuts [217] allowed users to select tools with voice commands in applications like Adobe's Photoshop. In this case, VoiceCuts would manage the selection of the tool as the user's vocal commands (in a *predictable* way), while not directly implementing on the artifact.

3.5.2.3 Within Technologies

Overall, technology types did not co-occur a lot (Figure 3.1c). This is principally because I focused on the ‘core’ technologies of the CST (often this was singular).

3.5.3 Cross-Taxonomy Analysis

I analyzed how CSTs have been designed by relating different taxonomies. My core research questions for this analysis are:

- How each *role* intersects with different *interaction approaches* and *technologies*?
- How each *user group* intersects with different *roles*, *interaction approaches*, and *technologies*?

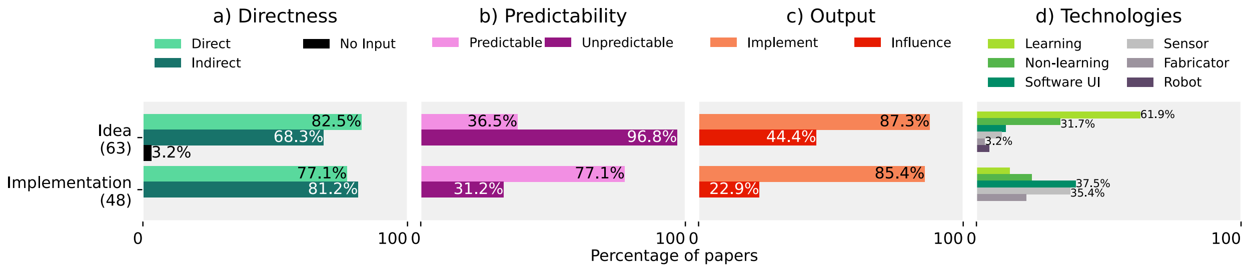


Figure 3.2: Percentage of interaction approaches (directness, predictability, and output) and technologies according to resource roles (items on the y-axis). On the y-axis, numbers in parenthesis indicate the number of CSTs with the code.

3.5.3.1 Role × Interaction and Technologies

Resource Roles, Interaction Approaches, and Technologies In Figure 3.2, I illustrate how *resource roles* relate to *interaction approaches* and *technologies*. For *directness* (Figure 3.2a), I found that *idea*-supporting tools slightly more often adopt *direct* inputs (82.5%) than *indirect* ones (68.3%). For *implementation*-supporting tools, there were more tools with *indirect* inputs (81.2%) than those with *direct* inputs (77.1%), but the difference was small. I also found that a small number of tools support *idea* without receiving any input from the user ($n = 2$, 3.2%). These tools function by prompting messages for inspiration without getting any input (e.g., “Tell me about your past creative experiences.”) [7, 193]. With the *predictability* (Figure 3.2b), I found that tools for *idea* tend to be more *unpredictable* (96.8% for *unpredictable*, 36.5% for *predictable*). Tools that generate arts to give inspirations to the users are one type of *unpredictable* tools that support with *idea* [69, 78, 79, 147, 197, 224, 304, 319, 399, 415]. Tools for *implementations* are shown to be more *predictable* (77.1% for *predictable*, 31.2% for *unpredictable*). One type of *predictable* tools for *implementations* augments the user's implementation actions while following the user's controls, like how paintbrushes are used on a canvas [67, 184, 186, 204, 368, 400, 444, 446]. Rarely, there were cases where a tool for *idea* is only designed with *predictable* approaches. For example, UnicrePaint [122] allowed a *predictable* but inspiring and unprecedented way of creating visual arts, stamping physical objects on a digital screen. There were also cases that the tool offers *implementation*-wise benefits only with *unpredictable* means. For example, in the fabrication of morphing material, SimuLearn [435] extends the user's *implementation* by informing the user of how the morphing would be done, which is *not predictable* to the user. With ways of how *outputs* are contributing (Figure 3.2c), I found that more tools are supported with *implementing* than *influencing* for both *idea*-supporting and *implementation*-supporting tools. This high occurrence of *implementation* across resources roles would be due to the prevalence of *implementation* among reviewed publications.

I also analyzed how resource roles and technologies relate (Figure 3.2d). For *idea*-supporting tools, learning (61.9%) and non-learning algorithms (31.7%) were the top two most used technologies. On the other hand, the top two technologies used for *implementations* were UI (37.5%), and sensors (35.4%). I give specific cases of technology use in each role in the next section, with process roles.

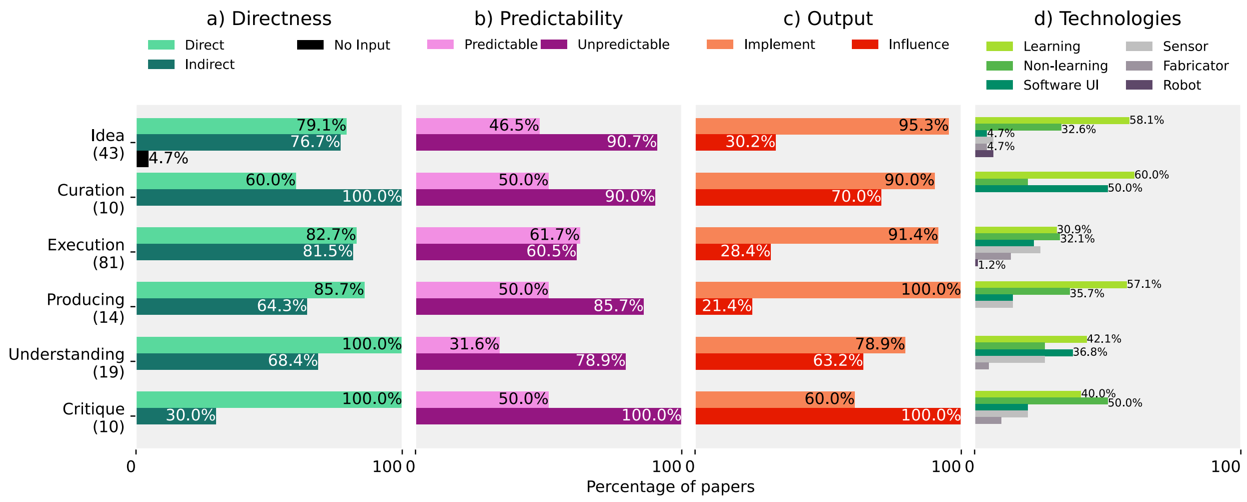


Figure 3.3: Percentage of interaction approaches (directness, predictability, and output) and technologies according to process roles (items on the y-axis). On the y-axis, the number of CSTs with the code is in the parenthesis. On the y axis, "Idea" stands *idea generation* and "Execution" is for *execution assistance*.

Process Roles, Interaction Approaches, and Technologies I analyzed how *process roles* intersect with *interaction approaches* and *technologies* (Figure 3.3). With input *directness* (Figure 3.3a), *direct* and *indirect* approaches are almost equally used in *idea generation* and *execution assistance*. For *curating* tools, all of them are supported with *indirect* inputs. Curating tools received natural language-based queries [117, 130, 219, 220], preferences [218], or partial information [428]. However, *indirect* inputs were not a necessary condition for *curation*, as some also received an artifact as a *direct* input [219, 426]. On the other hand, *producing*, *understanding*, and *critique* were more supported with *direct* inputs. Among them, all *understanding* and *critique* tools adopted *direct* inputs, as, by definition, they require artifacts to be understood [51, 113, 115, 171, 219, 220, 225, 237, 248, 253, 311, 383, 435] or evaluated [80, 94, 95, 230, 326, 355, 367]. Tools that receive no inputs are found only in *idea generation* (4.7%), which were robots prompting for inspiration [7, 193].

With *predictability* (Figure 3.3b), most *process roles* (excepting *execution assistance*) are more supported with *unpredictability*. Among these, all *critique* tools were *unpredictable*, as they give information that is unexpected by the user to change the user's behavior. Moreover, while most curation tools are unpredictable, there was a case of a predictable curation tool. This tool, Color Builder [368], allowed users to curate colors by arranging color swatches. By positioning color swatches, the tool would curate intermediate colors between swatches in gradients, which is predictable interactions to the user. On the other hand, *execution assistance* was equally supported by both *predictable* (61.7%) and *unpredictable* tools (60.5%).

For the *output* (Figure 3.3c), similar to *resources roles*, all *process roles* except *critique* are more enabled with *implementation*. Among these roles, *curation* and *understanding* used *influencing* more than others (70.0% and 63.2%, respectively). For *curation*, the curated artifact can be either included in the final artifact or used to influence users, like for inspirations [61, 130, 218, 219, 220]. For *understanding*, tools that *influence* often support by showing the analysis [51, 113, 219, 220, 225, 237, 248, 253, 311, 383]. On the other hand, tools that *implement* for *understanding* helped the user create an alternative representation of the artifact. For instance, Knotation [67] allowed choreographers to document choreographic processes into diagrams so that they can better *understand* them. Apart from these roles, all *producing* tools *implemented*, as the definition indicates. One thing to note is that there were cases where the tool is supporting *execution assistance* only through *influencing*. One type of such tool only augments the user's actions without directly manipulating artifacts [143, 431]. For example, when drawing with a pen, De-PENd guided the user's pen strokes with ferromagnetic forces [431]. I also found *critique* is more and all supported with *influencing* outputs, as it is to change the user's behavior.

With how *process roles* are supported by *technologies* (Figure 3.3d), *learning algorithms* were used frequently across different *process roles*. When I took a more specific look, *idea generation* was majorly supported with *learning algorithms* (58.1%), followed by *non-learning algorithms* (32.6%). These *algorithms* were often used to generate artifacts or information that can inspire users [57, 78, 131, 196, 243, 256, 273, 301, 320, 326, 356, 392]. Also, compared to other roles, *idea generation* had the highest percentage of using *robots*. These robot tools either prompt inspiring messages to the user [7, 193] or draw inspiring sketches on physical paper [243].

Curation frequently used *learning algorithms* (60.0%) and *software UI* (50.0%). *Learning algorithms* were often used to collect and curate materials [61, 117, 130, 218, 219, 220, 426, 428], while *software UI* were used to effectively present curated results [61, 117, 211, 220, 368]. *Curating* tools did not use *sensors*, *fabricators*, and *robots*.

Execution assistance showed the most distributed use of *technologies*, but was most supported by both types of *algorithms* (30.9% for *learning* and 32.1% for *non-learning*). Often, *execution assistance* tools that use learning algorithms generated a portion of an artifact [31, 31, 113, 147, 243, 256, 301, 415], like generating and adding a snippet of music upon the tune that the user have created [256]. However, still, for *execution assistance* CSTs, the majority of controls were on the users. Usually, the tool generates a small portion of the artifact, and the user could post-edit what the tool generated. These were the most frequently appearing type of designs for AI-driven CSTs that use generative algorithms. Moreover, compared to other *process roles*, the percentage of *fabrication* (13.6%) was highest in *execution assistance*. ThreadPlotter [155] is one example of enabling *execution assistance* with a *fabricator*, which allows users to create punch needle embroidery with X-Y plotter.

Producing was mainly powered by generative algorithms from *learning* (57.1%) and *non-learning algorithms* (35.7%). However, with generative algorithms, more tools have been devised with *execution assistance*. Different from generative *execution assistance* CSTs, *producing* tools allowed minimum post edits from the user. Instead, there were cases where it allowed continuous interactions through generation, like improvising music together [277].

Understanding used *learning algorithms* (42.1%) in the highest percentage, followed by *software UI* (36.8%), *non-learning algorithms* (26.3%), and *sensors* (26.3%). For *understanding* tools that analyze, *algorithms* were often used for the analysis [113, 219, 220, 225, 248, 253, 383], *UI* for the effective presentation of analysis [51, 113, 220, 237, 253, 311], and *sensors* to capture the artifacts [189, 253].

Critique frequently used *non-learning algorithms* (50.0%) followed by *learning ones* (40.0%). *Algorithms* generated *critiques* by analyzing artifacts [94, 95, 230, 326, 355, 367, 421, 422, 429].

3.5.3.2 User $\times$ Role, Interaction, and Technologies

Users and Roles For *resource roles* (Figure 3.4a), tools that support *both* levels of expertise and those for *specific population* had more tools that support with *implementations* than with *idea*. Other user groups were supported more by tools for *idea*. With *process roles* (Figure 3.4b), *novices* were supported with *critique* in a higher percentage (25.8%) than other groups. With

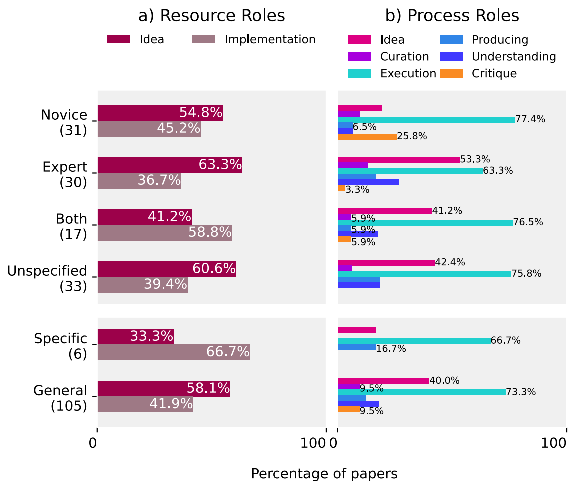


Figure 3.4: Percentage of roles according to user groups (items on the y-axis). On the y-axis, the number of CSTs with the code is in the parenthesis. In b), "Idea" stands for idea generation and "Execution" stands for execution assistance.

this role, novices frequently got critiques on how they can accomplish more high-quality creation with extended expertise [80, 94, 95, 367, 421]. However, *novices* were less supported with *understanding* (6.5%), compared to other user groups. I also found that *experts* are more supported with the role of *idea generation* (53.3%) compared to other user groups. This type of tool often allowed expert users to explore more possible options by generating one or more of them [131, 196, 197, 243, 254, 304, 315, 326, 362, 392]. For *specific population*, I found that they were not supported with *curation*, *understanding*, and *critique*.

Users and Interaction Approaches For *users* and *interaction approaches*, I found patterns with *novices* and *specific populations*. For the *directness* (Figure 3.5a), user groups other than *novices* and *specific populations* were similarly supported by *directness* and *indirectness* without large gaps ($\leq 10\%$ difference). On the other hand, *novices* and *specific populations* were more supported with *direct* inputs. With *specific populations*, I found that they had one case of receiving no input. It was a robot that prompts inspiring messages to children when creating titles for a drawing [7]. Even though there was one tool, due to a low number of tools for *specific populations* (n=6), it took 16.7%. For *predictability* (Figure 3.5b), the *specific populations* were the only group with more *predictable* tools, while other user groups were more with *unpredictable* tools. Those *predictable* tools for *specific populations* were designed to overcome their skill limits from not yet fully grown motor and cognitive skills (for children) or disability [55, 160, 245, 294, 441]. With *output* (Figure 3.5c), all user groups are more tend to be supported by *implementing* tools than *influencing* tools. Among them, *novices* were most supported with *influencing* compared to other user groups (48.4%), which is partly due to the high occurrence of *critique* tools in this population

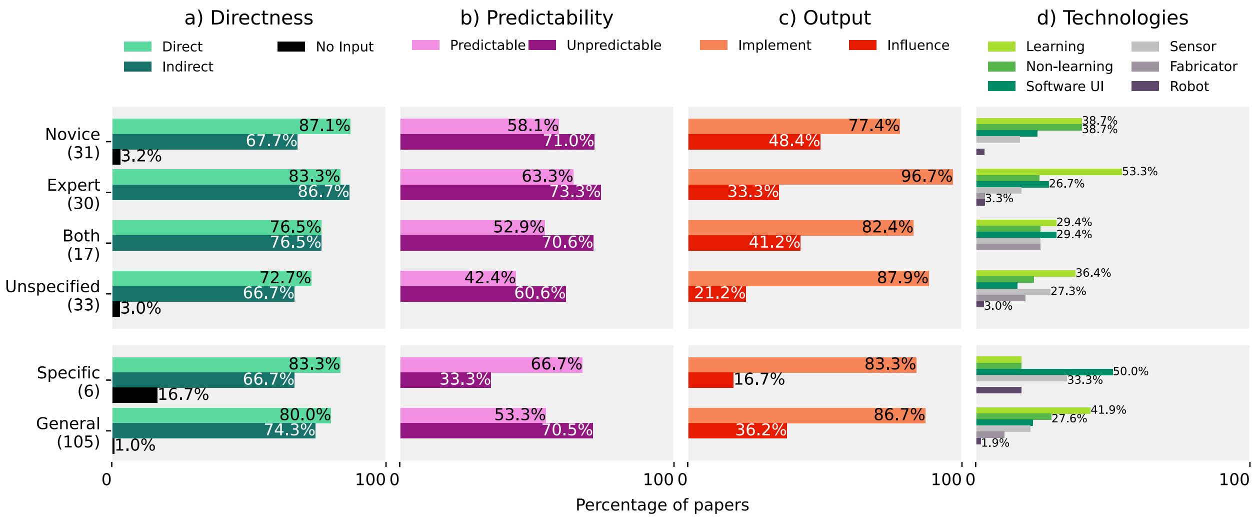


Figure 3.5: How each user group (on the y-axis) is supported with different interaction approaches (directness, predictability, and output) and technologies. On the y-axis, numbers in parenthesis indicate the number of CSTs with the code.

(which contribute through influencing).

Users and Technologies On analyzing *users* and *technologies*, I found that *novices* got more support with *non-learning algorithms* than other groups. This would be relevant to high support of *critique* roles in *novices*, as *critique* tools frequently use *non-learning algorithms*. Moreover, *novices* were not supported with *fabricators* or *robots*. Compared to other user groups, *experts* were more supported with *learning algorithms* (53.3%), which mainly powered *idea generation* tools. On the other hand, tools for *specific populations* showed the most distinct distribution of technologies. They used *UI* (50.0%), *sensors* (33.3%), and *robots* (16.7%) in higher percentages compared to other users.

3.6 Discussion

My analysis reveals interesting patterns in the construction of art-making CSTs from the HCI community. I first discuss how AI-CSTs have been designed, what can be their design issues, and how existing AI-CSTs tried to address them. I extend this discussion to identifying what is still limited with design existing approaches for general CSTs. Then, I hypothesize on possible reasons certain CST categories are more or less common.

3.6.1 Design Patterns of AI-CSTs and Their Limitations

With *resource roles*, *idea*-offering tools were more *unpredictable* and more powered by *learning algorithms* (see Figures 3.2). *Idea*-offering tool being *unpredictable* is expected, as contributing to artistic vision is more related to introducing inspiring ideas to users, which they could not have brought up themselves. *Learning algorithms* have been frequently used for *idea*-offering tools as they can find or generate novel ideas and artifacts. With *process roles*, we found that all roles except *execution assistance* tend to have more *unpredictable* tools than *predictable* ones. Moreover, process roles except *execution assistance* and *critique* have used *learning algorithm* mostly. This result resonates with findings from *resource roles*, as idea-offering tools tend to be *unpredictable* and use *learning algorithms*. Moreover, many idea-offering CSTs—except those for *critique*—tend to support through *implementation*. Furthermore, within *technologies*, *learning algorithms* were used the most. These findings give us insights into the design of AI-CSTs—that 1) *idea*-supporting tools frequently tend to be AI-CSTs, 2) while being *unpredictable* 3) and providing *implementations* at the same time. These insights add an interesting issue with the design of AI-CSTs, as unpredictable support that contributes to implementation with the mix of ideas can potentially intervene with the artist's vision and values within the artifact. This issue would likely intensify with AI-CSTs that leverage generative algorithms. For example, these AI-CSTs can potentially add distracting generations to what the artist has been creating.

When looked into generative AI-CSTs in reviewed papers, I could find these generating tools distribute among two *process roles*: *producing* (13 CSTs) and *execution assistance* (26 CSTs), with more CSTs in the latter. From this result, I saw that many CST researchers decided not to delegate all controls to these AI-CSTs. *Execution assistance* AI-CSTs are designed to allow a lot of user controls, from the specification on what to generate to post-editing. These design decisions would have been to defend the user's agency and sense of ownership while leveraging generative capabilities. Their design of controls, however, was limited to those that pre-specify what to generate or edit after the generation, not intervening in the generation process. Such a control approach would be enough to support gradual human creation when each unit of generation becomes a part of the created artifacts (e.g., an image asset in a game), only automating a part of the human creative process. However, such a design can introduce friction when the algorithm's result automates multiple steps of human creation without allowing human interventions during the generation process. In such a case, users would struggle to iterate on what AI has generated. For example, revising a digital painting generated in an end-to-end fashion would be more challenging than fixing a piece generated with multiple steps, as the user can intervene between those steps before the final rendition. Inspired by this issue, in Chapter 7, I introduce the design that allows users to steer AI generation during the generation process.

With producing tools, we found that some CSTs' interactions were more similar to those that people have with other humans. For example, some CSTs played a character in interactive narratives [164] or a participant chatbot in a radio comedy show [317], which was designed to mimic the interaction between humans (even though they are not perfect). Another example was a machine drum improviser that works with human performers [277]. More technical advances would render these types of human-like CSTs move viable. Hence, preparing for the future, it would be helpful to learn how interactions should be designed for these AI-CSTs.

3.6.2 Trends in CST Research for Art-making

The HCI community has built certain types of tools more than others. CSTs with *execution assistance* and *implementing* were those which were dominant in *process roles* and *output approaches*, respectively. I consider these as an extension of conventional art-making tools (e.g., paintbrushes). As these conventional tools are more familiar, CST researchers might focus on these design. It may simply be reasonable to imagine the translation of physical/conventional tools to digital versions. Another interesting observation in our data is that more tools support *idea* than *implementations*. I also observed that *idea generation* was the second most frequent role in the *process* codes. One possible explanation is that idea CSTs are more broadly applicable.

Regarding the design of interactions, many tools turned complex implementation interactions into simpler and more efficient ones. I could frequently observe this for tools that receive indirection inputs (41 of them). For example, Joinery [448] simplified complex laser-cut joint making with high-level parameters. Similarly, BodyAvatar [446] turned 3D modeling (which can require specific expertise) into simple gestures. This design approach would also likely apply to the design of future CSTs, including AI-CSTs, which allow users to control many different aspects. In Chapter 4, I discuss how this design approach would benefit the design of steering interactions for AI-CSTs.

We could also find that many tools used direct and indirect inputs together. As varying input modalities serve different purposes, adopting both input approaches would allow various expressions when using CSTs. For AI-CSTs, this combination would be more important, as users would give high-level steering specifications with indirect input while providing the materials that AI can work on as direct input. Moreover, users would like to edit the AI results with direct input. Hence, the design approach of mixing different input approaches would potentially facilitate the use of AI-CSTs.

Many CST papers did *not specify users*. Instead, they focused on describing opportunities introduced by new technologies and interactions. As a research community, I argue that CST researchers should be more deliberate in specifying the intended use. This will become increasingly important with novel AI technologies. Such tools are known to have significant biases. More specific acknowledgment of how a CST will be used and by whom may be vital in identifying and addressing unintended consequences.

3.6.3 How Roles Are Supported with Interaction Approaches and Technologies

While design patterns of idea-based, unpredictable, learning tools have been discussed, there are other patterns in how roles intersect with interaction approaches and technologies. First, I found that *implementation*-offering tools were more *predictable* (see Figures 3.2 and 3.3). Regarding technologies, *implementation*-offering tools more used *software UI*, *sensor*, and *fabricator*, which give better interfacing or means of construction. With *process roles*, *execution assistance* is equally supported with *predictable* and *unpredictable* approaches. Moreover, different *process roles* diverged along *directness* and *output*. *Critique* was the most noticeable among them, as *critique* CSTs are all designed to be *unpredictable*, *influencing*, and receiving *direct* inputs.

These findings reveal patterns of how CSTs can be designed for different *roles* with *interaction approaches* and *technologies*. In some cases, combining *roles*, *interactions*, and *technologies* would be inherently restricted. For example, *producing* less frequently co-occurred with *execution assistance* than with other process roles. Even when they co-occurred, they were actually enabled by separate functions. Moreover, all *producing* tools were *implementing*. This pattern is due to the nature of *producing*: *producing* tools do the majority of implementations on behalf of the user.

However, other patterns might have arisen as researchers focused more on one way of building CSTs. For example, the *implementation*-supporting tools were more *predictable*. However, there were exceptions, like tools extending the user's *implementation* with *unpredictable* simulation [435]. This exception is the evidence that researchers have employed a narrow focus when designing CSTs, while other approaches are possible. Another example is the use of *robots*. In the HCI community, the robot was rarely used to build CSTs (n=3). Moreover, while *robots* can enable tools to create physical artifacts, there was only one case a robot was used for *implementation*. This is a limited focus of our community—outside of the HCI/computing community, artists have been devising *robots* that *implement* artifacts by themselves or through collaboration with humans [2, 377]. The HCI researchers would be able to do a similar exploration while pushing beyond what has been done by other communities.

Some seemingly impossible patterns would be more feasible with technological advances. *Understanding* and *critique* tools using *indirect* input is one example. Designers of these tools have assumed that they must have artifacts to be analyzed. However, technological advances are introducing more expressive ways of making *indirect* inputs [44, 331], and they would increase the needs and feasibility for *indirect* inputs to be analyzed. For example, advanced generation models can produce content with natural language prompts [44, 331], which should be well-designed to get the intended results. Hence, CSTs would be able to analyze or give feedback on the user's prompts, so that they can be improved to draw desirable results.

3.6.4 Patterns in Supporting Users

Regarding how *users* get support, I found a few interesting patterns. First, *novices* were more supported with *critique* compared to other user groups. While *critique* is also valuable to *experts* (e.g., artists get feedback from colleagues), the CSTs I surveyed rarely supported *experts*. This may be due to the technical feasibility of giving *critique* that meets the expectation of *experts*. When *experts* receive *critique*, they would expect not only implementation-wise *critiques* but also more subjective feedback. For example, one possible expert CST critique role might be to learn and predict how people interpret an art pieces. Such *critiques* would be more difficult for algorithms and machines to generate, as they tend to be subjective. On the other hand, *novices* would significantly benefit from *critiques* that can improve their expertise. These types of *critique* tools would be easier to be designed and programmed as they require less subjective decisions. Hence, one possible future direction would be expanding the user population of *critique* tools to *experts*. At the same time, *novices* were shown to be less supported with *idea generation* and *understanding*. This might be because CST designers assumed that novices might not try novel ideas or build complex artifacts. We would also be able to expand the range of tools for *novices*, supporting them with *idea generation* and *understanding*.

Another pattern I found was in *specific populations* CSTs. The few examples were *implementation*-offering, *predictable* tools. This might be because researchers designed tools that can fill in the gap of the skill from disability or not yet fully developed motor or cognitive skills (for children). However, this pattern also might be due to the low number of tools for this population (n=6). Hence, I argue that this population requires more attention from CST researchers. It would be beneficial for these populations to have a broader diversity of tools. For instance, devising tools that can effectively help ideation for people with sensory disability (e.g., mood boards for the visually impaired) can be a valuable but under-explored topic in CST research. To accelerate research in this thread, we would be able to learn from accessibility or children-related research in HCI.

3.6.5 Comparison to Previous Work

For some parts of our taxonomy (e.g., process roles and expertise), I build on previous work. However, with new papers, I modified these by merging, removing, and adding certain categories. For process roles, my taxonomy expands Frich et al. [119]'s taxonomy on the creative process (pre-ideation, idea generation, evaluation/critique, implementation, iteration, meta/project). Here, I excluded meta/project, a taxonomy on the management, as I focused on a single user's interaction with CSTs during the artifact creation itself. Iteration is excluded, as it largely included support for repeating other processes. For pre-ideation and idea generation, I considered them to broadly contribute to ideation and combined them within the *aiding ideation* category. Pre-ideation would be close to *curation* in our taxonomy. For the taxonomy of implementation, I tried to identify more specific implementation approaches, which can potentially have different interaction dynamics.

For my taxonomy of users based on expertise, I also built upon Frich et al. [119]. In addition to the novice/expert split offered in the earlier classification, I added the code *both*. In Frich et al., the code closest to *both* is "casual" but this did not feel appropriate to CSTs that could support both types of users. Frich et al. [119]'s *casual* more implies tools that are "easy to be used by broad users". However, I used *both*, to include cases that support both experts and novices by having *a low threshold and a high ceiling.*

3.6.6 Limitations and Future Work

I focused on surveying art-making CSTs from HCI and computing research. My motivation was to understand how researchers in technological, human-centered fields have been designing CSTs. However, CSTs can be, and likely are, designed and developed outside of the community. Independent artists often devise unique CSTs to realize their artistic vision with new computing technologies. For such a community, the design patterns in roles, interaction approaches, and technologies would be significantly different from what we observed. Hence, studying other communities to investigate the commonalities and differences can be future work.

Our sampling approach introduces another limitation. First, I focused on tools that support the making of aesthetic artifacts with a single user. To understand the broader design implications of CSTs, considering other CSTs would be valuable future work. Moreover, as I did not filter recent papers with citation or download counts, analyzed papers are more biased to recent work (the number of the published paper is increasing). One relevant future work can be adopting a sampling approach that balances publication across time and doing temporal analysis.

I coded each CST with the dimension of predictability, but only considering intended behaviors. The spectrum of predictability can be wider, including unpredictable behaviors like unexpected errors. I only considered intended behaviors, because many CST publications do not have full information on errors. Future work can consider this aspect of "acceptability to the user". Furthermore, a single tool makes scoped outputs, not all imaginable ones. Hence, we would also be able to consider "possibility by the tool", whether the CST can make a certain output or not. With these, researchers would be able to analyze CSTs more comprehensively with the user's expectation. They would distinguish various tool behaviors, including intended results, known errors, unknown errors, surprising but favorable results, desired but impossible behaviors, etc.

3.7 Conclusion

In this chapter, to investigate the design of AI-CSTs, I studied how the HCI community has constructed art-making CSTs. For this purpose, I considered the roles, users, interactions, and technologies of CSTs. My work adds a more design-centered, consolidating perspective to the understanding of art-making CSTs. I taxonomized and coded 111 publications on art-making CSTs. Using those codes, I identified design patterns and design space of art-making CSTs. I found that vision-offering CSTs frequently adopt learning-based AI algorithms while being unpredictable and contributing to implementations. This pattern can introduce potential friction in using AI-CSTs from their unpredictability. For the potential issues, having steerability has been considered as a solution. From the findings, I identify that efficient and multi-modal interactions would facilitate steering of AI-CSTs.

CHAPTER 4

Steerability in AI-powered Art-making Tools

Based on the findings of Chapters 2 and 3, I summarize the design lessons for AI-powered art-making tools. The core lesson is that we need to design steerability in AI-powered art-making to facilitate effectiveness, efficiency, and iterative use in interactions. I also found that mixing familiar input modalities can be an approach to design such a steering interaction. In this chapter, I describe steerability, why users need it, its requirement, and a potential approach to facilitate it.

4.1 Definition and Need for Steerability

In this dissertation, I define steerability as *a user's ability to communicate and express their artistic intentions to AI-powered art-making tools so that their algorithmic behaviors can align with the user's intentions.* I draw this concept from the findings of previous chapters, which are closely relevant to user needs. For example, the artist interviews emphasize that clear and iterative communication of the artist's intentions leads to successful relationships with intelligent agents. Similarly, the design of inputs (e.g., whether they are direct or not and if users can predict outputs with them) has been considered as core aspects in the design of CST interactions. Steerability has already been considered important for art-making tools, as art-making is fundamentally the activity of expressing one's intentions. However, it becomes more crucial for AI-powered tools due to the unpredictability of AI models. Ideally, the user should be able to balance the divergence of the AI models so that the surprises of the AI models would not go beyond the user's allowed boundary (Figure 4.1). For such a purpose, steerability needs to be more nuanced and complex to let users specify where the AI model can diverge and where the model should follow the user's expectations.

4.2 Requirements of Steerability for AI-CSTs

The basic requirement of steerability is to ensure *effectiveness*, whether steerability was successful in conveying the user's intention to AI models and generating results that align with the user's

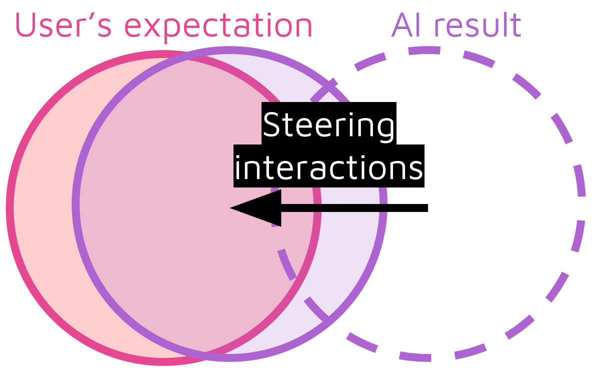


Figure 4.1: Steering interactions can bring AI results closer to the user's expectations.

intention. However, from the initial two studies, we identify two additional requirements: 1) facilitating *iteration*, and 2) *efficient* steering.

4.2.1 Facilitating Iterative Steering

One requirement is to facilitate iterative steering. Iterative steering can be composed of multiple steps: making steering specifications, observing intermediate results from AI models, respecifying, and repeating the process. It would be similar to how artists often work with other supporting actors, iteratively creating intermediate versions of artifacts and revising them after communicating about those intermediate versions. This repetitive communication and revision pattern ultimately leads to desired results. There are two reasons why steering interactions should support iterations. First, the language used for steering often tends to be under-constrained, requiring the user to iterate on it. Such under-constrained languages allow AI models to output "a range of results" that introduce surprises to the users. However, if the surprises are too misaligned with the user's intention, the user would need to iterate to align the result to their intentions. Second, in some instances, the user does not have a clear idea of what they want, but only realizes it after seeing intermediate results. If the realized desire does not match what AI models have generated, the user would need to iterate on steering.

4.2.2 Efficient Steering

Art-making is a complex process with many different aspects to specify (e.g., there can be many ways to fill in the blank canvas). Even creating one artifact requires skill and effort without involving another agent. While AI-powered tools can potentially automate and simplify many of the processes, they introduce their own large set of options, making steering specifications complex. While this challenge is universal for both non-AI and AI-powered art-making tools, previous

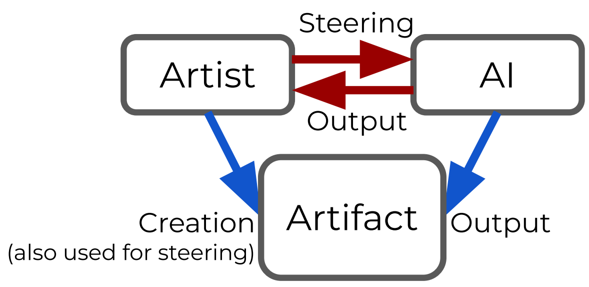


Figure 4.2: Dynamics of artist using AI tools

efforts in designing art-making CSTs have alleviated this challenge by adopting interactions that turn complex specifications into more efficient ones. Adopted strategies include high-level parameters [448] and sketching interactions [446]. Similar approaches to adopting efficient steering interactions would also be valuable for AI-powered art-making tools, as users would still have almost infinite options for the resulting creation and many complex details would need to be specified by the user. Note that the efficiency of steering interactions can be evaluated with several criteria, including time, the number of input interactions, and the cognitive effort that users perceived in generating the artifact.

4.3 Design Approach: Mixing Familiar Input Modalities

From the initial two studies, I propose a design approach to facilitate iteration, effectiveness, and efficiency in steering interactions, which mixes familiar input modalities. I identify this interaction approach by observing how artists communicate with supporting actors. When artists create art by themselves, they often use the modality of the art. For example, when artists paint, they use a visual modality, such as a paintbrush, and it is enough to create the visual arts. However, when artists have others helping them and need to direct their work on the artifact, artists need to provide guidance. It can take diverse forms, including 1) the combination of the artifact's modality and the modality other than the artifact's (e.g., marking up an image with annotations for visual arts), 2) only using the artifact's modality (e.g., a sketch of visual arts), and 3) only using the modality other than the artifact's (e.g., only text descriptions for visual arts). Among them, the use of mixed modalities has benefits, as these can complement each other. For example, verbal communication can be efficient as it does not require much effort to verbalize but can cause a mismatch in conception due to ambiguity. In such cases, references and sketches can help convey a concrete rendition, but they are often difficult to use flexibly. In such cases, using those means together would help artists to have more efficient and effective communication. Moreover, since each communication round can be efficient, it would also be easy to iterate.

In the context of AI-CSTs, using mixed familiar input modalities would have similar benefits, specifically when the interaction is already familiar to the artist. Even for general CSTs, as shown in Chapter 3, many tools used direct and indirect inputs together, implying the potential benefits of mixed inputs. Moreover, with recent AI technologies that can guide generations in the vector space or learn the multi-modality vector representations, there can be more opportunities in the design of multi-modal steering interactions. Specifically, AI-powered art-making tools can include mixed modalities in direct input to the artifact (blue arrow in Figure 4.2), indirect input to the AI tool (red arrow in Figure 4.2), or both. Adopting "familiar" modalities would facilitate effective and efficient steering, as users would easily learn how to use such interactions. In the later part of this dissertation, I introduce three AI-powered art-making tools that adopt steering interactions that mix familiar input modalities. The first is *Artinter*, which combines mood boarding and slider interactions to leverage the user's ambiguous and subjective language concepts as steerable handles for search and generation in visual art commissions. The second is *TaleBrush*, which allows efficient and iterative steering of story generation models with visual sketching interaction. The third tool is *PromptPaint*, which adopts the interactions of the paint mediums in the use of text-to-image generation models to facilitate efficient and iterative steering.

CHAPTER 5

Artinter: Steering Search and Generation with Mood Board and Slider Interactions in Visual Art Commission

5.1 Introduction

The commissioning of art pieces allows clients to request unique pieces suitable to their needs that at the same time retain the creativity, sensibility, and style of the artist [138]. Commissioned visual arts range from paintings to book and album covers and can be rendered in many different mediums. Because a commission is often 'bespoke', or unique to the client, clients have some influence over the final piece. That influence can vary from fine-level control of the content (e.g., requesting a specific portrait) to more limited input based on color, size, or where the piece will be exhibited. This does not necessarily mean that a client has absolute control of the final product. In many commissions, there is a necessary balance between the artist's agency to create their art and the client's needs or wants. Arriving at a mutual understanding is a complex process, as people do not share a clear mental model of each other's language, meanings, objectives, requirements, and skills [65, 96].

To accomplish a shared understanding of the artwork to be commissioned, communication is imperative [60, 65, 96, 312, 354, 402]. Artists and clients can have different languages due to subjectivity and expertise gaps. Successful communication often requires the creation of a common language that shares tacit conceptions, assumptions, and knowledge [96, 312, 402]. For instance, when the client asks for 'light' art, the artist and the client must develop an understanding of what that means. Is it bright colors? A specific "light" topic? Lighter brush strokes? Does it imply things that should be included? Or what should be avoided? A shared definition is critical to ensure that the artist's enactment of the concept can satisfy the client. Having achieved mutual intelligibility, artists and clients can ultimately converge on a commissioning 'contract,' a specification of the commissioned artifact. It can be as vague or specific as desired but ultimately defines the bounds of

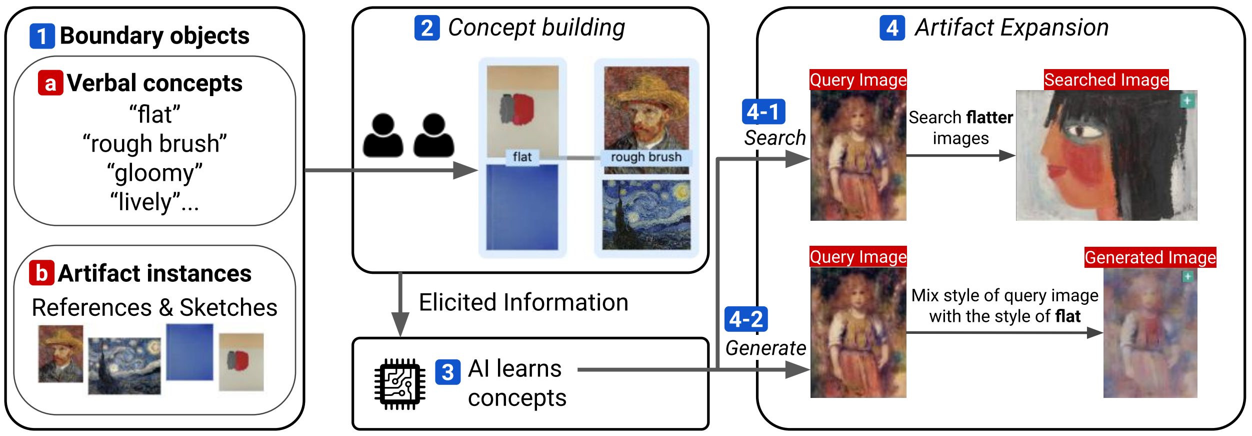


Figure 5.1: Design of *Artinter*. *Artinter* is a collaborative AI-powered system that supports communication around art commissions. It allows artists and clients to have shared boundary objects (1) on the mood board. With shared objects, *Artinter* allows users to have agreement on what they mean through concept building (2), or defining concepts with artifact instances. The information elicited from the development of concepts is also used to train *Artinter* to learn these user-defined concepts (3). Based on the learned concepts and shared artifacts, *Artinter* provides two AI-powered supports in expanding artifact instances (4): search and generation.

a good commission. With this contract in hand, the artist can focus on producing the commissioned piece in their medium of choice–digital or not.

In this work, we introduce *Artinter*, a synchronously collaborative AI-powered system. *Artinter* facilitates the use of boundary objects to help artists and clients share language and vision about the artifact to be created. The system aids this process by supporting collaborative sketching, art generation, and search. The features of *Artinter* were motivated through an analysis of guidelines and questionnaires for commissioning artists. My analysis identified various types of boundary objects [380, 381] that motivate the core of *Artinter*'s features. Specifically, I observed that different materials—language and visual—are used to convey information on the client's vision and the artist's description of their work (e.g., sample reference images, artist's vision statements, and client's goal description). As a commissioning relationship evolved, additional tools, such as 'mood boards,' would be leveraged to organize visual and language boundary objects [258, 259, 260, 278]. Through my analysis, I found common patterns among commission-centered boundary objects: 1) methods for describing *verbal concepts* (Figure 5.1-1-a)–natural language-based descriptions of artistic concepts, constraints, etc.; and 2) *artifact instances* (Figure 5.1-1-b)–concrete examples of imagery. For artifact instances, artists and clients created sketches [40] or searched for references [354]. In building *Artinter*, I focus on supporting both types of boundary objects through a unified interface inspired by mood boards—a space in which verbal concepts and artifact instances can be created, arranged, and organized to support mutual understanding.

Although boundary objects aim to reduce ambiguity, they cannot completely eliminate it [125, 234, 354]. Sometimes, such flexibility is desirable, as it gives the artist creative flexibility. When not, *Artinter* allows artists and clients to collaboratively define verbal concepts by identifying examples (e.g., "rough brush" from impressionist pieces vs. "rough brush" from action painting works) (Figure 5.1-2). Another challenge is that finding example artifacts can be difficult. Either the artist or client would need to sketch their idea–a time-consuming process–or search for examples. Introducing a natural language search engine to the mix adds more ambiguity as artists and clients must find a way to explain what kind of images they are looking for [219]. To address the challenge of finding or creating good examples, *Artinter* leverages existing user-defined concepts on the mood board as steerable handles. As users would have defined concepts and artifacts that center the communication, *Artinter* can learn them (Figure 5.1-3) and provide *artifact expansion*, AI-powered supports to explore more instances with artifacts and concepts within the communication context (Figure 5.1-4). Specifically, *Artinter* allows 1) guided search of other artists' works with user-defined concepts (Figure 5.1-4-1) and 2) generation of higher-fidelity sketches by combining concepts or shared artifacts (Figure 5.1-4-2), with simple slider interactions.

I evaluated *Artinter* in two studies: (1) synchronous art commission meeting sessions with six artist-client pairs; and (2) where 20 non-professional participants described a 'target' image with *Artinter*. From studies, I found that *Artinter* could help the artist and client users ground their communication means with mood board interactions while supporting finding references within their communication contexts with sliders of user-defined concepts. Moreover, I could observe that steerability from user-defined concepts facilitate iterative search and generation, gradually reaching instances close to the user's vision to communicate. Moreover, the generative AI functions helped users to accelerate creating high-fidelity sketches while lowering the required expertise bars. I also describe limitations of AI-based systems such as *Artinter*. These include the high effort to teach the AI some concepts, the AI not learning in the way the user expected, or users becoming overly focused on limited aspects of the specification when using generation functions. Based on these findings, I discuss how we can build future AI tools for art commission communication support with rapidly advancing AI technologies.

This chapter makes a number of contributions. First, I perform a preliminary analysis and find design goals for the unique process of art commissions. Based on these findings, I create *Artinter*, a novel collaborative AI-powered system that supports 'contract' communication during commissioning. Finally, I conduct two user studies with practicing artists and potential clients. My studies identify effective use patterns and features, as well as limitations. The results have implications beyond art commissions to other creative tasks including those with 'unbalanced' human-human collaborations. With *Artinter* as an example, I consider how focusing AI features on sub-tasks (e.g., aiding human-human collaboration) can produce better experiences over models that treat the AI as an 'artist replacement.' This chapter shows how steering interactions of mixed familiar input modalities can be used to align AI-CST behaviors to the user's contexts and facilitate iterative steering.

5.2 Background and Related Work

In analyzing related work, I reflect on the need for boundary objects in art-making communication and two mechanisms for getting them: search and generation.

5.2.1 Artistic Communication Means as Boundary Objects

While art production is often regarded through the myth of the 'lone creative genius,' the reality is quite different. Artists collaborate with many others: clients, gallery owners, agents, collaborators, etc. [25, 65, 190, 350]. Communication becomes a critical feature for success in these collaborative processes [40, 60, 65, 262, 263, 267, 354]. Communication helps to concretize and share tacit vision and knowledge with collaborators [96, 402]. Through communication, collaborators ultimately co-build a shared explorative language framework [312].

References [65, 354] are one key mechanism by which collaborators can concretely show concepts relevant to vision. *Sketches* are an alternative that allows artists to show their imagined vision [40, 65]. Various tools enable collaboration. For example, mood boards, which have a root in personal use (e.g., helping an artist track their own vision), are utilized in collaborative settings. Mood boards allow for the projection of one's vision into the collage of reference images and materials [258, 259, 260, 278]. The materials on the board, individually and collectively, act to concretize ambiguous or tacit ideas [278] and shape common languages and mutual understanding [259, 278]. Moreover, they can facilitate collaborators to explore more ideas [101].

All these mechanisms–sketches, references, mood boards, etc.–come to act as boundary objects [234, 380, 381]. That is, they serve as translations between different social worlds, satisfying the informational requirements for each of them. Boundary objects are effective intermediaries, reflecting commonalities for the shared understanding while allowing flexible interpretations for diverse participants. However, boundary objects tend to be unstandardized during the early phases of construction, causing disagreement in their understanding [234]. Thus, participants usually coordinate the definitions of boundary objects during their use. My view is that these characteristics of boundary objects coincide with artistic communication means [312, 354]. The design for *Artinter* focuses on facilitating boundary objects in art commission settings. Through the features of *Artinter*, we can investigate ways to support communication between artists and clients with these varied boundary objects.

5.2.2 Tools for Searching Artifact Instances

The references and sketches that form boundary objects need to be ‘obtained’ in some way–usually through search or generation. A client can try to find examples they like. This can be from the artist’s past work or the work of others. Similarly, the artist can use found examples to describe some idea or probe the fit of an idea to the client’s needs or wants.

Though they may not have been created specifically for the commissioning process, there are a number of search tools that may be useful. A common query language for these tools is simple keywords (e.g., Google Images). This approach is effective when the user can explain their needs or wants in natural language. However, in many cases, knowing the right keywords can be challenging [220]. Moreover, those right keywords might not map to either the artist’s or the client’s language. Introducing yet a third language can make search inefficient.

An alternative to text is abstract representations or visualizations that can be used to explore or move through image ‘spaces.’ For example, Dream Lens helps users explore many generative designs by visualizing them with various attributes and concepts [275]. It is a powerful approach in the hands of an expert, who has the skills to use digital tools or knows the ‘language’ of these tools. However, non-experts may find such tools difficult.

A third approach is to use example artifacts as input for exploration. Tools that support exploration by showing examples similar to user-given artifacts fall into this category [11, 107, 220]. Because users may want to find things that are ‘different’ from the example they provide, some tools seek to identify more divergent (e.g., serendipitous) examples [218, 219, 233, 341, 362]. These approaches can help find examples in the space ‘around’ an input vision. However, such tools tend to be personal in nature and do not externalize aspects that the user is looking for. Hence, these are not necessarily built to communicate concepts in this space.

Concept-driven tools are yet another alternative. These allow the user to explore artistic ideas by toggling ‘concepts’ as high-level parameters [47, 83, 222, 371]. For instance, by using the concept of ‘striped textures,’ the user can search for examples with or without striped textures. Unfortunately, the number of explorable concepts is often small as systems often require many annotated examples [56, 83, 222, 228, 300, 440]. An alternative is allowing individuals to define the concept independently with a small number of examples. Interactive machine learning (IML) tools often support this approach [10, 103, 112]. CueFlik, for example, allowed users to define complex rules (e.g., “product photos”) to find images [112]. These “few-shot learning”-based concept definitions became more plausible with powerful neural network representations [47, 146]. In *Artinter*, I follow this approach to allow users to configure concepts as a mechanism for defining complex queries.

I also note the utility and challenges of collaborative search. In the context of a group of people researching knowledge, such as working on joint projects or homework, previous work emphasized that sharing awareness, dividing labor without redundancy in efforts, and persistence in having a shared search session are crucial to facilitate collaborative search [12, 265, 284, 285, 286, 287, 313, 409]. Following these recommendations, *Artinter* supports collaborative searches of references with a shared mood board, synchronized user actions, and co-defined query languages.

5.2.3 Tools for Artifact Generation

While in many situations visual examples of ideas can be 'found,' sometimes they require 'creation.' This is important when creating is more expedient than finding but also when no example can be found. In these situations, collaborators resort to sketching or generating examples. Creating sketches requires skill and effort, which can introduce additional friction to efficient communication. For example, clients who feel like they cannot draw may resist creating a sketch. There have been diverse approaches to lower the expertise and skill bars in visual art creation by providing guidance and structure. However, this comes at the cost of limited flexibility in styles and content [28, 108, 183, 241, 387, 421, 429]. Recent approaches provide more flexible intelligent generative functions, such as co-creating drawings with the user [301] or generating images with the user's masking specifications [309] and natural language prompts [175, 177, 298, 330, 342, 348]. However, these do not necessarily consider the user's style, which can be a limitation in the art commission setting. Another set of techniques can generate images based on the styles of images within the user's inputs [123, 128, 307, 310]. *Artinter* utilizes a style transfer algorithm [128, 307, 310] to allow artists and clients to express artistic ideas with low effort and high fidelity while considering their styles or preferences. An individual can instantiate a rough idea by applying and mixing various styles including the artist's, and explore a space of possible artifacts.

As with collaborative search, there are also tools for collaborative artifact creation. These vary in sophistication from simpler sharing of artifacts [133, 149, 349], to co-creating on the same canvas [149], building upon each other's artifacts [133, 321, 447], and remixing multiple artifacts [321, 447]. *Artinter* builds upon these approaches to allow co-creating and combining styles of the artwork shared by collaborators.

5.3 The Art Commission Process

To understand the current practice of the art commission I begin by studying the materials artists currently use. Specifically, I identified two common types of materials–questionnaires and

	Code	Q (22)	G (27)	Examples
Target	Specify subject	68.2%	85.2%	things to be drawn, story, mood, personality
	Specify style	90.9%	55.6%	color, material, form
Means	Use text/language	100.0%	100.0%	textual inputs in questionnaires, conversation
	Use references	45.5%	59.3%	photo of subjects to be drawn, or any references that can be relevant
	Use sketches	N/A	44.4%	early sketch before the full development of the art piece
Approaches	Consider the artist's own style	31.8%	74.1%	choose preferred arts from the artist's previous works, the client does not dictate
	Language-only descriptions do not help	N/A	7.4%	verbal descriptions can be interpreted in diverse ways
	Periodic updates	N/A	63.0%	updating the client after a certain process and getting feedback

Table 5.1: Preliminary analysis results. Q stands for questionnaires and G stands for guidelines. The number in the parenthesis indicates how many questionnaires and guidelines are analyzed.

guidelines–that can be used to better model the practice. Many artists who accept commissions use structured questionnaires to obtain information from clients. The second source, guidelines or training materials, is information content for artists who are starting or developing their commissioning practice. These are often in the form of blogs or social media posts and are most often found in artist-focused sites and blogs. Taken together, these materials reveal the dynamics of the art commission process and how artists view the process. Note that their focus is more on artists and might have limitations in understanding client perspectives. To obtain a large collection of these, I searched Google for keywords such as: "art commission questionnaire", "guidelines for art commissions" and "tips for art commissions." By excluding search results that are not art commission questionnaires or guidelines, I collected 22 questionnaires and 27 guideline documents. Their domains ranged from fine arts to sculpting, illustration, and cartoon creation. For questionnaires and guidelines, one of the authors did iterative coding with inductive analysis, and another author reviewed codes.

5.3.1 Findings

I focus the discussion on codes related to communicating artistic ideas as they have the most implications for *Artinter* (summarized in Table 5.1). Most questionnaires and guidelines had some focus on obtaining a high-level description of the *target* of the commission. This fell broadly into two categories: *subject* (a thing to be drawn) and *style* (constraints of the art including the color, material, form, etc.). Specific questions about *target* range from concrete (e.g., objects to be drawn) to ambiguous (e.g., the story of the piece, overall mood, or client personality). In many situations, artists also used this information to determine if the client relationship would work. Thus, a function of the initial commissioning discussion might be to recognize if a commission is possible or not. For all cases, *targets* were communicated through *text/language*, either with textual inputs or verbal meetings. I found that some artists additionally use, or request, artifact instances such as *references* and *sketches*. *References* were usually visual materials that would be used as models of a piece (e.g., a photo of a subject), effectively conveying what the client wants. Guidelines on interacting with clients suggested creating collections of *references* (e.g., through a mood board) or developing less costly *sketches*.

Another point of emphasis in the training materials was finding the balance in keeping an artist's own style while allowing clients to choose limited aspects to control. In practice, this was approached in a number of ways. For example, some artists explicitly ask clients to choose references from the artists' collection work. This managed the client's expectations and constrained them to things the artist could or would produce. A few guides mentioned that using only verbal descriptions would confuse communication, as they can be interpreted in many different ways. Notably, the broader idea of 'management' manifested in other pieces of advice. For example, advice sites suggested avoiding clients who would micromanage. The advice also included the management of expectations by periodically updating the client so that they can track the progress and give feedback. Clearly, the need to establish and maintain boundaries on what *will not* be done (i.e., what's 'outside the fence') is as important as what *will* be produced and how.

5.3.2 Design Goals

As our core interest was to build a communicative medium between the artist and the client, the most interesting findings were the use of textual/verbal means and artifact examples as boundary objects. Based on the analysis and previous work, I identify challenges and design goals to support the use of boundary objects in art commissions. I would emphasize that art commissioning is a complex process and there are other aspects to consider than facilitating the use of boundary objects. As a simple example, the need to periodically update clients on progress. These are outside the scope of the current prototype, but I discuss them as potential future work in Section 5.7.4.

First, in artistic domains, boundary objects can be highly ambiguous and interpreted in diverse ways when a single modality is used. My analysis revealed this limitation in verbal descriptions (e.g., 'what does it mean when they paint with a rough brush?') [65, 354]. Artifact instances would complement verbal descriptions (e.g., 'here's an example of what I mean by rough brush'). However, using instances without text would bring in a similar ambiguity problem [125] (e.g., 'did they like the color or brush technique of this piece?'). Hence, **textual descriptions and artifact instances need to be scoped and connected as necessary (D1)**. I emphasize the *as necessary* part of this design goal as the tool does not have to provide a translation dictionary or mental model mapping more than needed by the artist or client to specify the commission. Overspecification of boundary objects can narrow the artist's creative freedom and prevent the client from seeing pleasant surprises in commission results (if they want them) [65].

My second observation is that while *having* reference images is great for a commission, *finding* reference images is often costly. For example, the client may not know how to describe what they're looking for. In such a case, they will struggle with a search tool that doesn't map to their concepts [83, 222, 371] and queries [220]. Similarity-based search tools [11, 107, 220, 362] may become too disconnected from queries or intents. For example, the client may not want many images that look the same (e.g., many images with the same color sunset) but rather a set that describes the boundaries of that concept (e.g., a range of what would acceptably fit in the concept of the sunset). By observing the interaction patterns between client and artist, one can more readily identify what it is that they are searching for [65]. Hence, **artists and clients will benefit if the guided reference search conforms to their communication contexts (D2)**. Such conformity would also allow artists to scope the searched references to those relevant to their style, helping them keep their art styles.

Finally, for many commissions, there may not be a perfect reference image. If there is one in the artist's style, the client would likely just buy that. Thus, more unique sketches of the proposed piece may serve as a way of resolving both the meaning of certain language constructs (e.g., 'let me draw what I mean by figurative') or the even bounding the final artifact (e.g., 'here's a rough sketch of what I'll paint'). Sketches would be a flexible communication means for artists with the expertise to produce them. Unfortunately, sketches can be costly to create. For example, a sketch with more detail may require more effort (e.g., textured sketches would take more time than simple line drawings). Moreover, sketching requires expertise, preventing non-expert clients from creating one. While creating a sketch close to the artist's style would facilitate communication, that would be even more difficult for non-expert clients. To address these issues, **sketches should be created quickly, with details, with low expertise, and within the style relevant to the artist's scope (D3)**.

5.4 Commission Communication with *Artinter*

Motivated by the design goals, I built *Artinter* (Figure 5.2). *Artinter* is designed to support art commission communication by grounding and expanding the use of boundary objects. *Artinter* is composed of two parts, a mood board on the right and a sketch pad on the left. On the mood board, users can share artifact instances and verbal concepts. Within this space, examples can be grouped, linked, and labeled collaboratively. The sketch pad allows users to quickly instantiate new ideas. In the mood board, users can perform *concept building* (Figure 5.1-2), collaboratively defining verbal concepts with artifact instances to ground these as agreed communication means (D1). Using these defined concepts, *Artinter* also allows users to perform *artifact expansion* with two functions: searching references with concepts as handles (D2) or generating sketches by combining concepts and existing artifacts (D3) (Figure 5.1-4). I designed the search and generation to center around

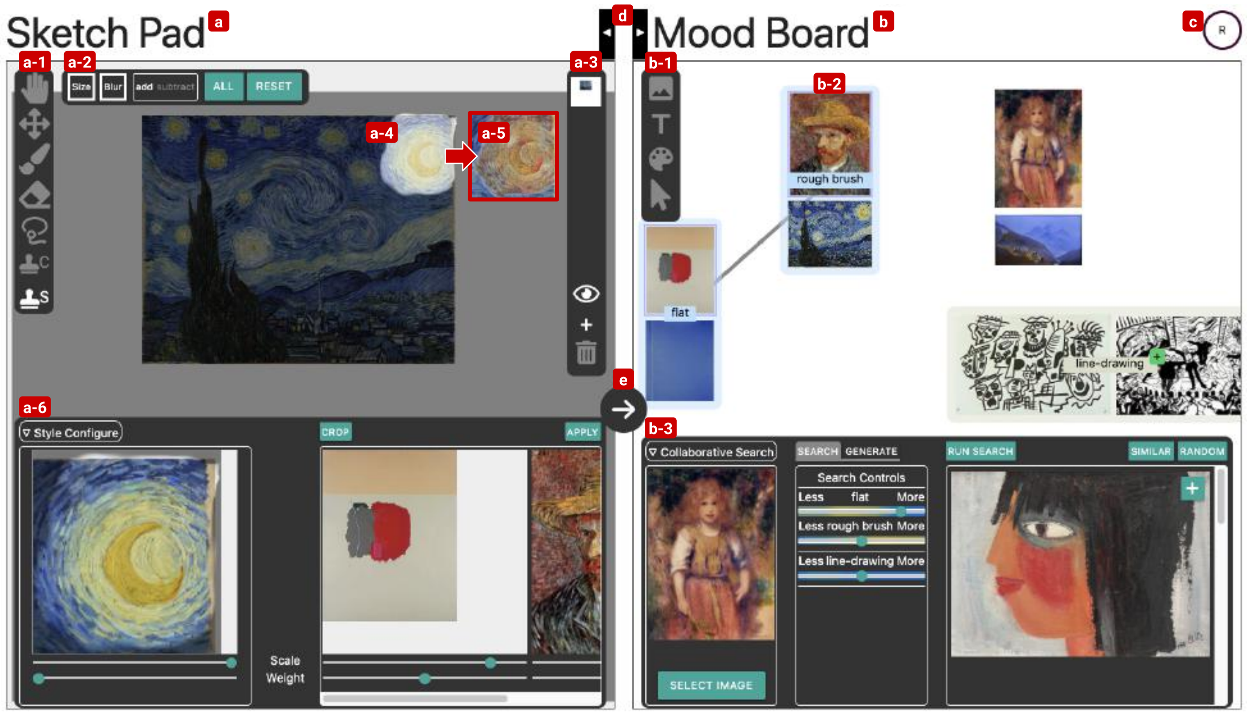


Figure 5.2: *Artinter* interface. *Artinter* is composed of a sketch pad (a) and a mood board (b). On the sketch pad, users can do raster image editing, such as brushing, erasing, or lassoing (a-1). Users also can copy-paste images on the mood board onto the sketch pad. With layers, users can add, remove, reorder, and hide them (a-3). On the mood board, users can add art images, texts, and color swatches (b-1). Note that color swatches are considered similar to art images except that they have a single color and the color is selectable. With art images or color swatches, users can define concepts (b-2). On the sketch pad, the user can also combine styles of images on the mood board and apply them to one of the layers (a-2, 4, 5, and 6). The example result of the style mix is presented in a-5. On the mood board, users can collaboratively search or generate artifact instances with concepts defined by themselves (b-3). They can also know who is online on the boards (c). The user also can expand a certain panel if they want to focus on working on one (d). The current image on the sketch pad can be moved to the mood board (e).

the user-defined concepts to align these supports with human communication. To better explain the features of *Artinter*, I describe the system with an interaction between Juno, an oil painter, and Louis, a client. Juno and Louis are artist and client personas derived from my preliminary study and I use these personas to capture how *Artinter* would help art commission processes.

5.4.1 Set-up

Louis wants a piece expressing the concept of "isolation" in Juno's style. While he likes Juno's style, he doesn't yet know how his idea can be represented. Similarly, Juno thinks the theme of isolation may manifest in her recent winter landscapes that utilize what she thinks of as 'rough brushing.' However, Juno does not necessarily know if Louis' idea of isolation aligns with hers or if Louis will understand her use of certain technical terms. Louis and Juno decided to use *Artinter* synchronously in a remote meeting session to get support in arriving at an agreement (or ‘contract’) on what artifact to create. Note that *Artinter* focuses on supporting style-specific factors in visual arts (e.g., colors, textures, patterns, etc.). These are more ambiguous and difficult to communicate than other factors like forms or subjects.

5.4.2 Concept Building (D1)

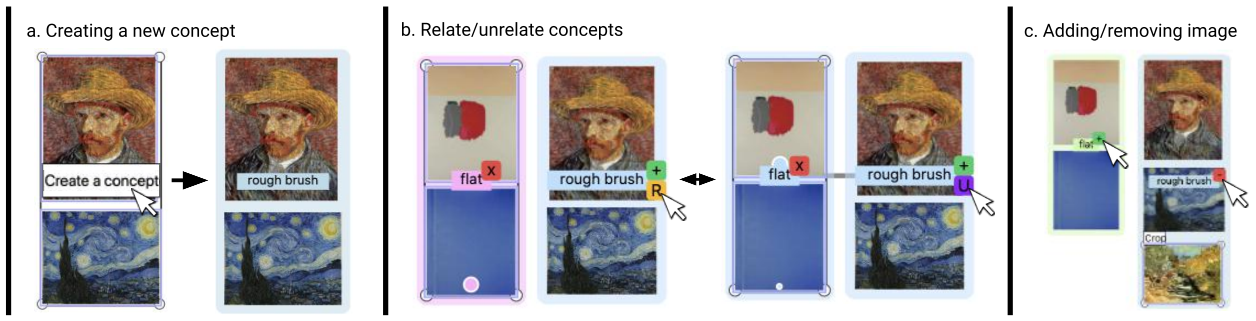


Figure 5.3: Building and updating concepts in *Artinter*. a) A user can create a concept by clicking the CREATE A CONCEPT button and inputting a textual name for the concept. b) While a user has selected a concept, the user can relate the concept to other ones by clicking R buttons showing up on other concepts (left). The user can also ‘unrelate’ already related concepts by clicking U buttons on concepts connected to the selected concept (right). c) The user can add a selected image (bottom-right) to a concept that does not have the image by clicking + button (left). It also can be removed from a concept that already includes the image by clicking - button (right).

Juno and Louis first use *Artinter* to try and become mutually intelligible about each other’s languages and scope of interests. It is necessary to remove ambiguity in communication and identify the boundary of what each other would be interested in. For this, Juno can load examples of her artwork into *Artinter*. She organizes works according to her own classifications by moving and grouping them visually on the mood board. For example, many of Juno’s pieces have a distinctive brushing technique. She can select images with this rough texture and click CREATE A CONCEPT button to create a group with the name of “rough brush” (Figure 5.3a). With the creation of the concept, a box with color will visually encompass images in the concept with its name displayed on it. The design of this concept-building interaction is partially inspired by card sorting [378], where people group pieces of information under categories.

After creating the group, Juno realizes that Louis is still unclear about what an alternative to the “rough brush” style might look like. She starts to think that other concepts can be semantically related to “rough brush”. She imagines that an antonym, such as the “flat texture” of watercolors might better clarify the concept for Louis and makes a group for that concept. She also adds a few concepts for adjacent oil painting styles (e.g., other brush techniques) that are hypernyms to the rough brush concept. To indicate the relation between these, Juno can visually connect concepts. This serves two purposes: (1) it contrasts and highlights a concept's meaning in relation to other concepts for the human participants; (2) it further trains *Artinter* to be more accurate by contrasting relevant concepts to each other. When Juno selects the concept of "rough brush", 'connector buttons' will appear on other concept groups (Figure 5.3b). When the concepts are not yet related, an R button shows up, which (r)elates, or connects, them. If they are already related, a U button shows up, which (u)nrelates, or disconnects, concepts. When two concepts are connected, they are connected with a line and have the same background color.

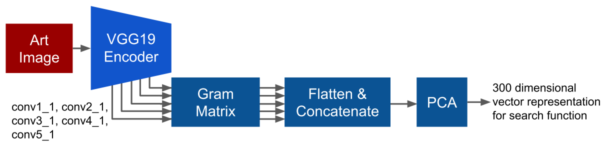

Art Image
VGG19 Encoder
conv1_1, conv2_1, conv3_1, conv4_1, conv5_1
Gram Matrix
Flatten & Concatenate
PCA
300 dimensional vector representation for search function

Figure 5.4: Pipeline for acquiring vector representation for recognition and search function.

As Juno defines concepts with example references, *Artinter* learns to recognize patterns, modeling the user language. Internally, *Artinter* creates vector representations for images and concepts. It uses two types of representations, one for recognition and search, and another for generation function: search requires a low number of dimensions (for quick search), while generation requires a high number of dimensions (to model style differences). For the representation for recognition and search, *Artinter* takes intermediate layers of the VGG19 encoder [372], which are relevant to style elements in visual arts [128] (Figure 5.4). Then, *Artinter* takes the gram matrix of each layer to flatten and concatenate them. Because this vector tends to be huge (610304 dimensions), *Artinter* reduces them to 300 dimensions by training a PCA algorithm. For the artwork dataset used in PCA training, I used WikiArt dataset [419]. With this representation, *Artinter* trains a linear classifier for each concept, with negative examples coming from 20 randomly sampled art images in the WikiArt dataset [419]. When more than one concept is related, instead of having multiple classifiers for different concepts, *Artinter* trains a single classifier with multiple classes. Learning on multiple concepts would allow *Artinter* to contrast different concepts and accurately learn them. Critically, this concept-relating function also impacts the performance of the search. I report on the performance of this recognition pipeline in Appendix A.1.1 and include specific examples that can impact *Artinter*'s performance in Appendix A.2. I return to the details of the sketch generation vectors below.

As the user adds new images, *Artinter* labels the examples added on the board based on how *it* modeled these concepts. A potential side effect is that we can help users understand how *Artinter* learned concepts and if it matches with how users think of those concepts. Using this feature,

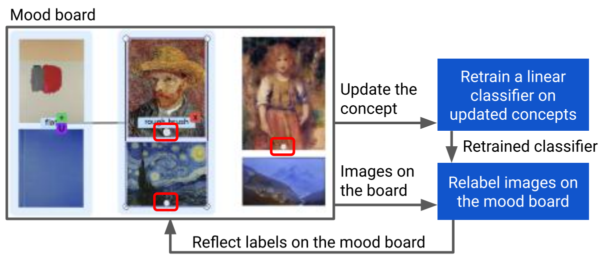

Mood board
rough brush
Update the concept
Retrain a linear classifier on updated concepts
Retrained classifier
Images on the board
Relabel images on the mood board
Reflect labels on the mood board

Figure 5.5: Iterative process of updating concepts in *Artinter*.

Louis clicks on the ‘rough’ concept and observes how *Artinter* annotates all the images on the board. *Artinter* shows a small circle label on each image which indicates that *Artinter* recognizes the concept from the image (red boxes in Figure 5.5). Circle size encodes the confidence of the algorithm that the image relates to the concept. If *Artinter* does not recognize the concept in an image, it does not show the circle. Louis can also see labels for each image by selecting an image and placing a mouse over it. As Louis learns the concept and how *Artinter* learned them, he can start to collaborate with Juno on including other adequate examples into the concept. This could be to broaden or narrow the concept’s scope (based on Juno and Louis’ discussions) but can also serve to improve erroneous models. With examples added or removed from the concept, *Artinter* updates its model of concepts by retraining the classifier. (Figure 5.5) After retraining, *Artinter* creates updated labels for images on the board. Such a model update also happens when Louis and Juno relate or unrelate concepts. If Louis does not understand Juno’s concept well, *Artinter*’s labels can help them capture it. For example, at one moment, Juno can see images labeled with the concept while she does not think so. Juno and Louis chat about how to define the concept and iterate on the concept based on their decision. Note that each iteration happened within a few seconds, allowing the interactive use of *Artinter*

5.4.3 Artifact Expansion

Louis begins to study Juno’s concepts and captures core style elements that comprise Juno’s artwork. While pondering on how “isolation” could be expressed with Juno’s style elements, Louis thinks that the artwork to be rendered would need to have characteristics a bit different from Juno’s style to that point. He likes the idea of the rough brush. However, he thinks isolation is better captured through a more abstract image, something potentially articulated by an even more extreme form of rough brushing. Juno is open to this idea. For this, Louis and Juno decide to use artifact expansion functions—search and generation—to acquire additional artifact instances such as images and sketches. In *Artinter*, Louis and Juno can tie this process closely with the communication context, as they can use shared concepts and artifacts as the means for steering expansion (e.g., generate a piece that has the artist's concept of *a gloomy winter, empty winter landscape* while having *rougher brush texture than the artist's*"). Using concepts and artifacts, Louis and Juno can either extrapolate (e.g., find examples with rougher brushes than what Juno has) or interpolate (e.g., mixing references from Louis and previous work from Juno) from existing artifacts. Hence, newly found artifacts would be relevant to Juno's unique style while expanding to what Louis is interested in—clearly drawing the boundary of their agreed scope.

5.4.3.1 Search with Concepts (D2)

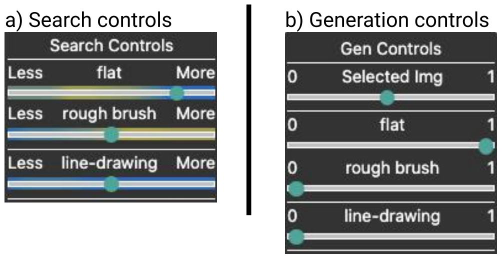


Figure 5.6: Concept controls for search and generation function.

Juno and Louis decided to collaboratively search for additional references (Figure 5.2b-3). They first set the query image, which is the start point of the search (the first column in Figure 5.2b-3). For instance, they can select Juno's artwork that is considered the most similar to what Louis wants. They can "search with concepts" as the search interface shows each defined concept as the search control slider with each end indicating LESS and MORE (Figure 5.6a). This control allows users to decide how much more or less of the concept needs to be considered in the search, compared to the query image. For instance, to find references with a flatter texture than Juno's works, Louis can set the slider value for "flat" higher. Sliders for related concepts are under the same partitioned area (e.g., "flat" and "rough brush" in Figure 5.6). For added emphasis, each slider shows color gradients, which indicates whether the results will be similar or different from the query image. This visualization is for the estimation of results before actual querying. Yellow indicates that more similar images will be retrieved, while blue signals that the output will be far different. With multiple concepts, when one slider value changes, other sliders update color gradients, as different sets of images will be searched. Louis and Juno can see search results by clicking on the RUN SEARCH button (top-left of the third column of Figure 5.2b-3). As they find useful search results, they can add them to the mood board for further discussion (+ button in the third column of Figure 5.2b-3). Juno and Louis also can search for images similar to the query image or random ones (top-right of Figure 5.2b-3). *Artinter* can be designed to allow for additional search interfaces (e.g., keyword driven).

Our concept-based search pipeline is similar to Cai et al. [47]'s approach. While *Artinter* learns a concept, it calculates a concept activation vector (CAV) [208], which represents the directional vector of the concept in the ML representation space. When running a search, *Artinter* embeds the query image into the vector representation, and *Artinter* adds the CAV of each concept with weights from sliders. From the calculated vector, *Artinter* retrieves the nearest neighbors from external datasets (the WikiArt dataset in the current prototype). A technical evaluation of this search functionality is in the Appendix A.1.2.

5.4.3.2 Generation by Combining Concepts or References (D3)

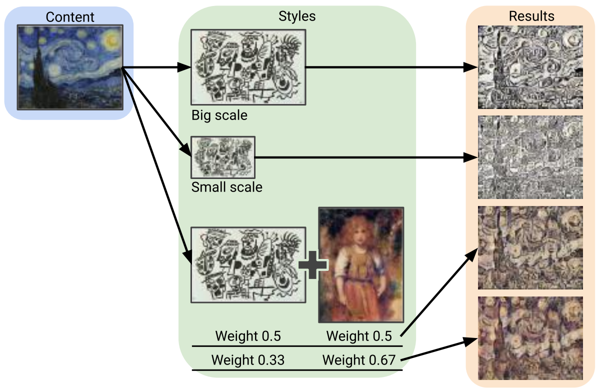


Figure 5.7: Impact of scale and weight parameters in sketch generation function on a sketch pad.

With new examples from the search, Juno clearly understands what Louis wants. However, they still don't know if it will work with her style. Juno can use *Artinter*'s generation function to mix her styles with those of found references (Figure 5.2a-2, 4, 5, and 6). This function gives a quick preview of a piece without full-fledged creation, helping Louis and Juno to more easily agree upon which direction they will pursue. The recommended way of using generated examples is as references. However, for digital art these could be used as a part of the final artifact. However, such inclusion or use can present challenges to the artist with their sense of ownership of the created art piece. When the final piece is to be physically created (e.g., as an oil painting), the dynamics around the sense of ownership may be different. I return to these issues in Section 5.7.3.

Artinter allows generation in two modes: the first is on the sketch pad (Figure 5.2a-2, 4, 5, and 6). With the "style-stamp" function, Louis and Juno can first define the "content"–the area of the sketch pad layer to apply the style mix. They can do this by drawing the area on the sketch pad (e.g., a small area of the image as shown in Figure 5.2a-4). Alternatively, they can click the ALL button to choose all areas with the image content in the layer. The selected area will show up in the first column of the style configuration panel (Figure 5.2a-6). For "styles", Louis and Juno can select images or concepts from the mood board. They can select one or more. If a concept, rather than an image, is chosen, all images under the concept will be selected. *Artinter* shows chosen images in the second column of the style configuration panel (Figure 5.2a-6). For each style and content, Louis and Juno can control two parameters: the first is weight, how much weight a specific style will be given when mixing styles. The second is scale, how large or small the style will be applied to the content. The art image with the chosen scale is shown in the interface to help users estimate how big the style will be applied. Figure 5.7 shows how these parameters impact the generation and more cases are included in the Appendix A.3. Louis and Juno also can crop a part of a style image to use only the part as the style. The process of generating style mix with style-stamp is not synchronized between users to minimize interference from actions of multiple users (e.g., avoid two users simultaneously manipulating the same layer). *Artinter* shows only the generated results (Figure 5.2a-5) to all users.

The second mode of generation happens on the mood board. This mode is for quick style combination with simpler controls, instead of giving all fine-grained controls. This mode is collaborative, with manipulations of controls synchronized between users. This function can be used in the collaborative search panel (Figure 5.2b-3) by clicking the GENERATE tab on the top of the second column. The generation controls would show up in the second column (Figure 5.6b). For simplicity, Louis and Juno can only control weights for concepts, but not other aspects like scale and cropped areas. The query image serves as the content, and users can also assign a weight for the query image (Selected Image in Figure 5.6b). When the user clicks the RUN GENERATION button (which replaces the RUN SEARCH button in Figure 5.2b-3), *Artinter* generates the image with the mix of styles in the third column.

A style transfer algorithm, SANet [307], powers the generation. For this, *Artinter* used vector representations based on VGG19 [372]. The mix of styles is done by linearly interpolating styles with weights. The scaling of style is achieved by scaling input images according to the user-defined ratio and then getting the center crop of each style image with the smallest dimension of all style images. To "learn" the styles of concepts for the generation function on the mood board, I averaged all style representations of images in the concept. I chose this approach as I designed generation to be the function that interpolates. Because I used an existing style transfer algorithm, I did not technically evaluate it [307].

After collecting boundary objects necessary to specify the art to create, including textual notes, references, and examples, Juno can start working on her physical canvas.

5.4.4 Implementation

Artinter is implemented as a web application, using HTML, CSS, and Javascript. I used React as a front-end framework and Feathers as a back-end framework. Feathers enabled real-time synchronization of user interactions. I had a separate server to handle machine learning calculations, implemented with Python and Flask. For the CAV model, I modified Tensorflow-based TCAV codes[1]. For the style transfer algorithm, I revised SANet code[2] implemented in PyTorch.

5.5 Study 1: Support of Design Goals

In this user study, I aimed to learn how *Artinter* supports clients and artists regarding three design goals identified in Section 5.3. Specifically, I focus on how the core features of *Artinter*–concept building and artifact expansion–achieve these goals in synchronous and iterative communication settings. To answer this question, I conducted an observational study with a follow-up interview.

5.5.1 Participants

I conducted the study with six artist-client pairs. I recruited artists with commission experience from Upwork and social media advertisements (two females and four males, ages 19-36, M=27.5, SD=6.4). They had one to 11 years of experience in visual arts (M=6.5, SD=3.7). Five artists had more than ten commission experiences, while one had four commissions. Their styles fell into two main types: abstract art (3) and illustration (3). I recruited clients from our university mailing list (five female and one male, ages 26-56, M=39.7, SD=12.8). Two of the client participants had experience with art commissions. I compensated each artist $82 and each client $25. In discussion, I code sessions as S1 to S6, artists as A1 to A6, and clients as C1 to C6.

5.5.2 Procedure

I designed our study to simulate the initial meeting for commission. I matched artists and clients based on time availability. If the client did not like the artist's style, I matched them with another artist. It was to avoid the client not being engaged in the art commission process. I asked participants to watch a tutorial video and try *Artinter* individually before the scheduled session to familiarize themselves with the tool. To minimize the interference of this pre-session access to the study, they were not allowed to interact with other people with the tool.

[1]https://github.com/tensorflow/tcav
[2]https://github.com/GlebBrykin/SANET

I invited the artist and the client to a remote art commission meeting. I conducted the meeting on Zoom and recorded it. I first reminded them about the main features of the tool and how they can use those features. During the session, I first asked the artist to build a board by defining the core concepts of their art. I guided them by prompting that concepts can 1) distinguish their artwork from other people's (i.e., artist's unique concepts) or 2) characterize differences within their artwork set (i.e., different concepts within the spectrum of the artist's style). I also asked the artist to verbally explain concepts to the client so that they could build shared languages.

After building the mood board, I asked the client and the artist to discuss which artwork the client wanted the artist to work on. I mentioned that their specific goal is to bring up artwork specs that can satisfy the client. Constraining the task, I asked them to focus the discussion on the visual styles. After the client and artist agreed on what to create (constrained to a maximum of 35 minutes), I asked them to do a survey and an interview on how they used the features of *Artinter*.

5.5.3 Results

Our findings are based on an analysis of the survey results, video recordings, and interview data. For qualitative data, I did iterative coding with inductive analysis, and a collaborator reviewed them. I focused on identifying how the functions of *Artinter* support the three design goals from Section 5.3. In addition to relating to design goals, I also report diverse usage patterns and potential limitations of *Artinter*. Note that when I append count to qualitative findings, that indicate the number of sessions where the findings were mentioned.

5.5.3.1 Survey Result

For the survey (Figure 5.8), artists and clients were asked different sets of questions about whether *Artinter* helped users 1) communicate artistic ideas generally (Q1s, Q2s), 2) have shared languages (Q3s), 3) search references (Q4s), 4) generate examples (Q5s), and 5) consider the artist's style (Q6s, Q7s). The participants' perception was overall positive. Specifically, overall positive responses of Q3s, Q4s, and Q5s indicate that most of participants perceived that *Artinter* supports its design goals. In the later sections, I present *how Artinter* could support those design goals. Only one client participant answered negatively, that they could not effectively test out and communicate ideas with style mix generation. I expand on this case in Section 5.5.3.6.

5.5.3.2 D1: Concept building

Artinter supports D1 if it can help participants mutually understand the language they are using in relation to images. The survey result broadly support user perception that this was the case. Additionally, I present qualitative observations.

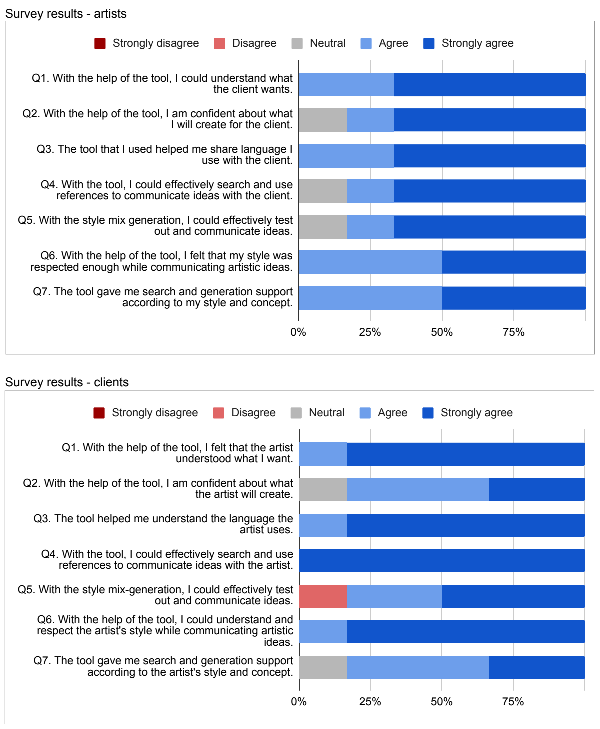


Figure 5.8: Study 1 survey results.

Verbal concepts and artifact instances are complemented by each other with concept building and participants defined concepts flexibly during the session. Participants mentioned that concept building externalizes concepts with example artifacts while explaining artifacts effectively with concepts (N=5). For example, C2 mentioned that *"the concept of group, the similar pictures together, is kind of use of a verbal way that translates the picture to a verbal (concept)."* While I prompted artists to build concepts about their artwork, I could also observe that some participants worked to define/concretize client preferences (N=2). One example from S2 (bottom-left of Figure 5.9) shows a proactive use of concepts and images by the client. Moreover, participants grounded their concepts based on the communication contexts, which enabled them to efficiently convey nuanced and complex concepts efficiently. For example, participants decided on concept names based on the context of the conversation between the artist and client (N=4). Hence, it was sometimes hard to estimate the concept's visual styles only with the names. For example, in Figure 5.9, the artist and client defined a concept of illumination on a single object. As all images in the concept had chairs as the main object, they named the concept "chair", making it difficult to grasp visual styles only with the name.

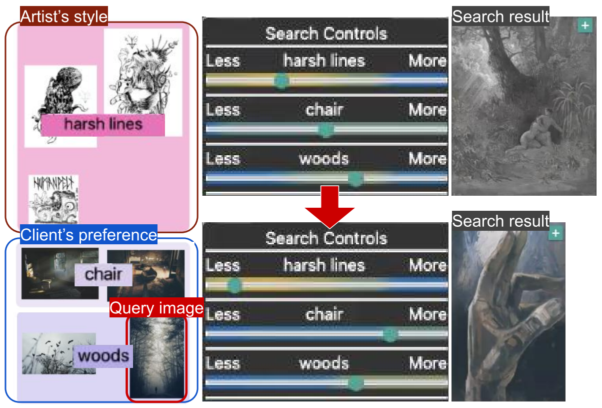


Figure 5.9: Participants defined concepts that can explain the artists' styles or clients' preferences. Participants could also collaboratively search for references with envisioned style-wise characteristics by adjusting sliders for user-defined concepts ("harsh lines", "chair", and "woods"). Artwork under "harsh lines" are by Jesse Hughes (A2), and those under "chair" and "woods" are provided by the C2.

5.5.3.3 D2: Artifact Expansion-Search

Given D2, I expected that *Artinter* would allow users to search references within their communication contexts (e.g., by leveraging terms they use for communication). This was observed in a number of ways.

User-defined concepts enabled an iterative steerable search. In search, iteration in concept-based search control helped participants capture what the counterpart wants within their communication contexts (N=3). For example, in Figure 5.9, A2 and C2 first searched the image at the top right. C2 wanted an image with more diverse colors and illuminations on one object but thought the first image did not have enough colors due to the "harsh lines." Hence, C2 decreased the value of "harsh lines" for the next search. At the same time, they increased the value for "chair", as they thought the ambient textures of included images would contribute to finding images with illumination on one object. As a result of the search, they could find the artwork at the bottom-right. By comparing the color combinations between two search results, A2 was able to better understand what colors C2 wants. In a case like this, participants could use concepts from their communication context (e.g., "chair") seamlessly in the search, guiding its usage. Participants stated that the search results helped them clearly communicate ideas (N=3). Some even inspired participants (N=2). A3 referred to this as, *"stirring the pot."*

5.5.3.4 D3: Artifact Expansion-Generation

Artinter supports D3 by allowing participants to create high-fidelity sketch images of relevant styles in a short time and with a low effort. Sketch images generated in such a way would support

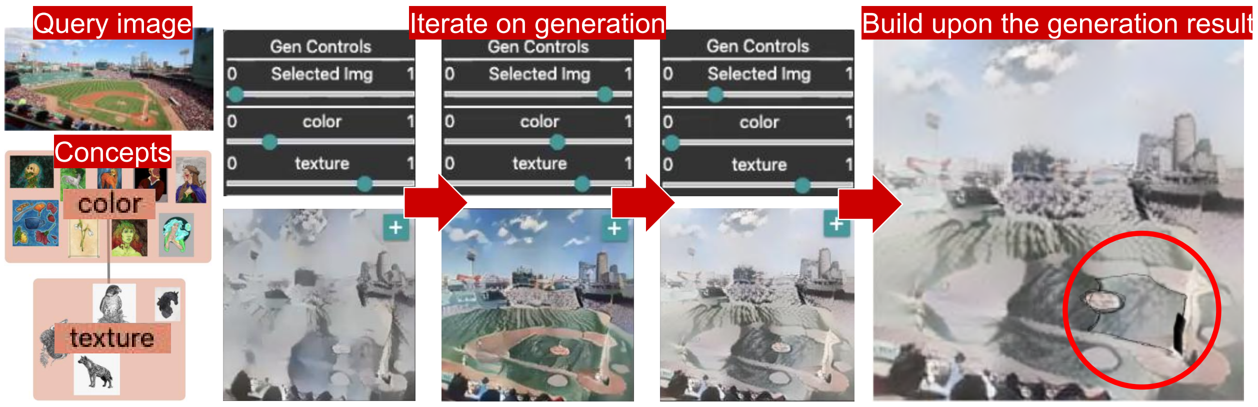


Figure 5.10: Participants iterated on the generation by controlling available parameters and reached the desired one. Participants built upon intermediate or final generation results by pointing out preferred characteristics or adding sketches (as in the red circle). Artwork under concepts are by Danielle Moses (A1), and C1 provided the query image.

communication between users.

Iteratively controlled generation shows the participant's desired specifications. Participants indicated that generation could capture the styles in artifacts or concepts, mix them, and apply them to other images to express their artistic ideas (N=4). In addition, participants mentioned that they could make more targeted adjustments with controls such as weights, scales, and areas to crop (N=3). These controls allowed them to find *"good in-between of the different things"* that *"went in line with what the user was thinking"* (A2). Moreover, I could observe that the *differences between* the iteratively generated results could show the user's desired specifications (N=4). For example, in Figure 5.10, by iterating on the weights of each concept for generation, A1 and C1 could reach the result close to what they envisioned. By comparing results generated within iterations, participants could realize how much definite contour and colorfulness the client participant wants. Participants also mentioned that the generated results could inspire them (N=2).

Generated results served as a "starting point" for further discussion. In all sessions, participants verbally communicated about generated results or built upon them (e.g., by adding sketches). They used this approach to explain details not presented in the generated results. The rightmost image of Figure 5.10 is one example, where A1 sketched on the generated result to convey that A1 will draw outlines to separate areas.

The generation function can increase the efficiency in communicating and exploring ideas. Usually, coming up with a sketch takes time and effort. Hence, exploring diverse options would be difficult. As the generation function shrinks the cost of experimenting with ideas to a few clicks, participants anticipate trying out more ideas and investing more time in stages other than ideation or communication (N=3). For example, A2 mentioned: *"If I was a painter and was doing some-*

thing (a high-fidelity sketch) in this style, I would probably still take a week from there, but getting to that point without me having to spend five hours, it's like I'm throwing things with a pencil... that's a minute. You can't really do better than that."

The generation function lowered the social barrier in expressing preferences on the artist's style. In one interesting session (S3), when the generated art that mimics an artist's style does not directly belong to the artist, clients could more comfortably show their preferences for the artist's styles with the generated artwork. C3 noted: *"It's easier to say to criticize the work of generated art than saying, 'Hey A3, the piece number X that is in front of you, that's terrible. And piece Y is great!"'*

Artists have different perceptions about the generation function "mimicking" the artist's style. A3 mentioned that they were somehow 'scared' of the generation function, as the algorithm accurately mimicked A3's style. It could have been relevant to their fear of AI technologies replacing their occupation. On the other hand, A2 was not as uncomfortable with this process and mentioned that even human artists mimic each other and get inspiration from others: *"Even if you could have the most unique art style in the world, but you still looked at things and had thoughts about things, and that influenced what you do in some way."* A2 also perceived the generation function more as a part of the tool under the user's control. I further discuss its implication in Section 5.7.3.

5.5.3.5 Diverse usage patterns of *Artinter*

Participants used *Artinter* in different ways, showing that their goals and processes can vary.

Clients engaged in the discussion to different degrees. In most cases, the artist took more control of the tool than the clients, artists suggesting ideas with *Artinter*, and clients mostly giving feedback (N=4). However, when clients wanted to realize a specific style, they expressed their opinions more strongly, even participating in the search or generating artifacts (N=2). For example, in Figure 5.9, C2 actively added references that show their preference and expressed their opinions while choosing search parameters (e.g., adding 'chair' in the iteration of the search). C6 thought that *Artinter* helped with more engagement, allowing *"not just communication, but almost collaboration."*

Artists varied in how frequently they used the search function. If the artist was open to trying out diverse styles or the client's desired style that is different from the artist's, they tend to use the search function more actively (S2). When the artist did not use the search function (S1), they said they did so because they already have their well-defined styles. It indicates how much artists would want to extrapolate from their style varies between cases.

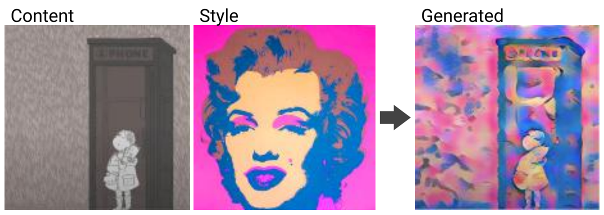


Figure 5.11: The case of the user's mental model not aligning with the AI's result. A5 expected the generation function to apply uniformly flat coloring as the style image, but the result showed a bit of a mix of colors in the background and other parts of the image. The image on the Content is by Rory Lucey (A5).

5.5.3.6 Limitations of *Artinter*

Generation function on the sketch pad was not collaborative enough. While I strived to implement collaborative features that would support simultaneous work, these features were still limited. For example, the sketchpad did not show how the counterpart is using the generation function. For cases where the client and the artist wanted to discuss how to use the generation on the sketch pad side, it could be a limitation (S2, whose client's response to Q6 in the survey of Figure 5.8 was negative). I believe I can alleviate this in future versions of *Artinter* by sharing the use of generation on the sketch pad.

User's mental model of how the generation works might not accord with how it works. A5 mentioned that the generation function did not behave as they had assumed. For example, in Figure 5.11, A5 wanted the generated result to have uniformly flat coloring in each area of the drawing, but *Artinter* rather mixed the colors with gradients. To counter unexpected generation results, A5 wanted demonstrations, explanations, or even more controls to help them better understand and manipulate the generation function. Additional scaffolding (training, tutorials, streamlined interfaces) may address this concern.

An infrequently used feature. In three sessions, participants did not relate concepts together, which could result in non-optimal use of recognition and search. The participants might have simply forgotten about the function due to the complexity in learning the tool or might not have found relating concepts natural. A mixed-initiative approach may help here, such as *Artinter* suggesting concept relationships. It would serve both to bootstrap this kind of label and remind users of this feature.

5.6 Study 2: Usage Patterns

Study 1 identified diverse usage patterns and limitations. Because there were three 'agents' at work–the artist, client, and AI–it was difficult to specifically isolate the human-AI interactions. To better understand the interaction dynamics (and limits) of each user with *Artinter*, I conducted a second study where I qualitatively examine the use of *Artinter* compared to that of the baseline version tool, without concept-building and artifact expansion features.

5.6.1 Participants

From university mailing lists, I recruited 20 participants (eight female, 11 male, and one gender variant/non-conforming, ages 19-33, M=23.2, SD=4.0). A few participants were the client in our first study. While I was not specifically seeking to find artists to participate, a number of participants had experience in visual arts. Nine participants had prior experience with asking for or commissioning artwork from an artist or designer. This recruitment approach was motivated by the goal of understanding how users with different backgrounds might utilize *Artinter* and our AI features. I compensated participants with a gift card worth $20. I code participants as P1 to P20 in the later part of this chapter.

5.6.2 Procedure

I conducted an observational study with two conditions: 1) *Artinter* and 2) *Baseline*, a version of sketchpad and mood board without AI features. To focus on studying the dynamics between each user and AI functions, Study 2 was done as an individual, asynchronous exercise. Specifically, I gave participants a 'target' art piece with the task of creating a mood board that describes/explains the target to someone else. This partially models a situation where a client might have a strong vision in mind when commissioning a piece.

I constructed target images synthetically. I chose this approach because participants could easily explain existing art pieces by stating the name of the art piece or the artist (e.g., "recreate Van Gogh's Starry Night"). I synthesized the images with SANet [307] by combining realistic photos as content and sampled artwork as styles (e.g., a photograph of a street market mixed with the painted style of a specific artist). For style images, I selected five random artists from the WikiArt dataset and sampled two works from each artist. It allowed me to generate multiple possible targets. Sampled art pieces were all 2D paintings in physical mediums, such as oil paintings.

I conducted the study over Zoom. After a brief overview of the study, participants went through two sessions—*Artinter* and *Baseline* conditions—which were randomized in order. For each session, participants watched tutorial videos of the tool. Videos were split into each group of functions (e.g., a video for search functions). After each video, I asked participants to try the functions introduced in the video. I presented the video segments describing AI functions in the first or second session based on which condition they went with first (i.e., right before they did the *Artinter* session).

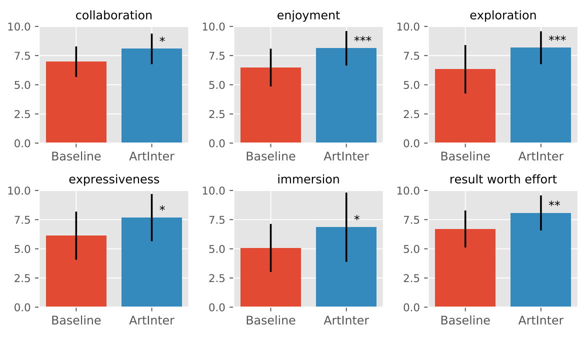


Figure 5.12: Creativity Support Index results. *, **, and *** indicate significant difference with $p < 0.05$, $p < 0.01$, $p < 0.005$, respectively. The error bar indicates the standard deviation.

Because the system has many features, training took about 40 minutes in total. Once completed, participants were given the target art piece and asked to build the mood board. I told them that the mood board should act as a 'specification' of the target to an artist. I also mentioned that they could not use the target art piece on the board or as input to functions (e.g., they were not allowed to use the target art image as a query for the search feature). In both conditions, participants were allowed to use Google Search to find an initial set of reference images. Participants had a maximum of 20 minutes to build a mood board. After each condition, I asked participants to complete a Creativity Support Index survey [59]. The survey attempts to measure how well the tool supports the creative activity on a scale of 0 to 10. After participants finished both sessions, I conducted a short semi-structured interview with them. I asked questions about their experience with the tool and how their strategies varied in two conditions. In total, the study took approximately 100 minutes. These studies were recorded and transcribed.

5.6.3 Results

5.6.3.1 Creativity Support Index Results

I found that participants perceived *Artinter* to be more effective in performing tasks than *Baseline* (Figure 5.12). Specifically, when tested with Mann-Whitney U test, *Artinter* was perceived to be more collaborative ($p < 0.05$), enjoyable ($p < 0.005$), explorable ($p < 0.005$), expressive ($p < 0.05$), immersive ($p < 0.05$), and worth putting effort ($p < 0.01$).

5.6.3.2 Qualitative Results

I present qualitative analysis results that show specific usage patterns and limitations of *Artinter* in our study design. I conducted iterative coding with inductive analysis on video recordings and interview data and a collaborator reviewed generated codes. Note that for these results I focus on reporting those that either expand on insights from the first study or were not previously described.

Concept building-based functions require a steep learning curve and high user efforts. In many cases, concept-building helped participants define concepts related to the target while conveying their perception of concepts to *Artinter* (N=18). However, how *Artinter* learns concepts did not necessarily match all participants' ways of thinking about them (N=2). There was one case where the participant did not understand the concept-building mechanism at the beginning (P2). *Artinter* considers overlapping visual characteristics of images in a concept group as what the concept represents. However, P2 understood that *Artinter* would pick different aspects of each image and mix them to form a concept. Due to this misunderstanding, P2 had two images with far different characteristics in a concept, expecting AI to combine the texture of one and the color of the other as the concept. One participant (P11) also mentioned that the tool did not allow them to teach as they wanted. P11 noted that some concepts, such as the roughness of textures, are better specified with the spectrum but not with the discrete categories. Such a disagreement between user expectations and tool design could lead to friction when using the system.

Moreover, concept building required additional efforts from users. Similarly to what I found in the first study, the search function performing best with "Related" concepts was one source of friction. Among 20 participants, only eight participants made connections between concepts with the "Relate" function. Among those who made connections, only three participants made connections between meaningfully relevant concepts (e.g., warm colors and cool colors). Others connected concepts that were not necessarily relevant (e.g., "blurry" and "dull color", which are not on the same dimensions). Furthermore, the study design puts more workload on participants using *Artinter*. Unlike the first study, which resembles a realistic commission setting where the users would have the artist's works as initial inputs, the second study setting does not have initial inputs, requiring participants to collect examples by themselves. It could introduce friction if the concepts that the users are to discuss do not exist with any of the user's initial set of references. With more efforts and a steeper learning curve to use the system, *Artinter* could have shown only a small benefit compared to the baseline condition.

The generation function was useful to mix artifact instances and concepts, extending the ability to express intents. It was perceived positively, as participants could bring up a new artifact close to the target by mixing different concepts and images (N=14). Participants also mentioned that *generation* allowed them to do more than what they could do in the *Baseline* tool, as they could flexibly "create" an image to express the visual characteristics of the target art (N=6). The participants thought this extended ability helped them accurately describe their artistic intentions.

Another specific opportunity I also observed in the first study was that the users tend to iterate on the generation by changing parameters and mixing (N=12). These generation results gradually reaching towards target arts could be helpful to convey the user's intentions more clearly. Specifically, the delta in style changes between iterations could help denote which characteristics the user wants to express. When using generation, however, some users tend to focus on replicating the limited aspects (N=3). For instance, users can be inaccurate with the color of generated results while more closely replicating the texture of the target. It was due to either the lack of time using the complex tool that the user just learned about or limitations of *Artinter* in separating style elements as the user wants. For instance, when example images under the concept "red" also have the attribute of "flat", the "flat" style can blend in when the participant uses the concept for the generation.

Participants diverged in how they used *generation*. Ten participants generated images by mixing different individual art pieces while five mixed concepts. Interestingly, two participants created their own "style patch" on the sketch pad and used that as one of the style images used in the *generation*. Three participants did not use generated images due to either unsatisfactory results or lack of time. Among those who generated images, six used a single "final" generated piece to indicate all aspects of the target image, while eleven others used generated images to denote partial aspects of the target image (e.g., only for textures).

5.7 Discussion

Artinter is the first prototype to support the commissioning process. From the user studies, I showed opportunities and limitations in using AI functions to support art commission communications. Specifically, *Artinter* adopted mixed input interaction modalities—mood board and slider interactions, in addition to visual sketching on the sketch pad—to allow users to effectively communicate their intentions. I discuss 1) mechanisms to ground concepts for steerability, 2) iteration as a means to communicate artistic ideas, 3) tensions in learning the artist's styles, and 4) AI-CSTs for unbalanced collaborative contexts.

5.7.1 Mechanisms to Ground Concepts

Concept building in the mood board interface is the mechanism to model the user's language. It could allow users to form their own steerable handles for search and generation. However, one challenge in concept building was that users tend to have varying expectations of how AI function learns boundary objects, sometimes disagreeing with how AI actually learns. I identified two patterns, one where the user did not yet have a good mental model of how to do concept building, and the other due to limitations of the tool in expressing concepts. For example, the user who assumed that the concept building would mix styles of different grouped images would be the first case, as they could not understand that a single concept should have art pieces with shared styles. Limitations to express ordinal or continuous dimensions with *Artinter* would be the example for the second case.

On the basis of the type of issues, I propose two different approaches. When users struggle to understand how AI learns, the tool can better guide users. For example, in addition to providing examples with which *Artinter* would learn well, providing failure examples would also help. On the other hand, when the tool has limited expressiveness in concept grounding, we can design interactions with more flexibility. It would require improvements in both interactions and algorithms. Flexible prompt-based approaches can be a potential direction. For example, as how we can flexibly prompt language models to serve a wide range of tasks [44, 329, 337], we can allow users to specify concepts more flexibly with the mixture of visual art examples and natural language descriptions (e.g., overall brush stroke direction is similar to the first art piece while the level of roughness is between the first and the second art piece) [123]. However, there can be potential issues with this approach: while what the user can input would be less limited, the model might not be able to handle all the prompts. Moreover, with more flexibility in grounding concepts, the machine's mechanism can be more opaque to the end-user [337]. Without addressing these challenges, introducing flexibility would hardly facilitate concept building.

Another challenge in building concepts was that users often do not relate relevant concepts even when doing so would result in more accurate recognition and search. Mixed-initiative approaches might potentially facilitate use of this feature. Creating relations between all concepts can be a simple option. However, this would not result in optimal use, as relating too many concepts can degrade system performance (see the technical evaluation in the supplementary materials). A better approach would be to consider user-defined concept names and identify potentially relevant concepts from them. While such an approach would be possible, it would still have limitations. For example, in our studies, users sometimes decide on the concept name based on the "communication context," but not on visual attributes of the concept (e.g., "chair" for images with lights illuminating on a single object, as the object tends to be a chair in given examples). In such cases, intelligent relation-building functions might fail to identify concepts that the user thinks as relevant. Hence, fully automated concept-relating might have limitations, and users would need to be allowed to make the final decision by themselves.

As *Artinter* learns concepts, the user would need to understand how it learned them. I expected that users would leverage *Artinter*'s recognition results to understand it. However, in our study, participants rarely mentioned recognition and instead seemed to rely on search/generation results (similar to people drawing references to indicate how they understand concepts). This could be due to the complexity of the tool or a learning curve issue (e.g., users may not even remember certain features exist). One relevant future work direction can be running a study on each function to investigate their specific utility and usability.

5.7.2 Iteration for Artistic Communication

From Study 1 and 2, I could see that results produced from the iterative use of AI functions could be boundary objects that clearly show the user's intentions. Here, comparing intermediate results and recognizing the delta between results were the core mechanism to reveal what participants want. For such a mechanism, the generation function would be more effective than the search, as iterative generation would only change limited aspects whereas the search would output far different results, with which the desired delta is difficult to recognize. However, not all generation algorithms would be adequate for this purpose. Only generation algorithms that support iterative and gradual changes would successfully support this purpose. Future work can potentially redesign *Artinter* to better leverage this iterative use of AI functions. For instance, I can design the system to track the iteratively generated artifacts with version controls, with visualized paths that the users explored.

5.7.3 Tension in Generative AI

Participants in the first study showed a spectrum of reactions about *Artinter* mimicking the user's style—from being fearful to being totally unconcerned. Artists may have recognized that the generated images are not good enough to serve as the final artwork. However, as generative algorithms improve, the distribution of reactions is likely to change. For instance, diffusion algorithms caught attention with the capability to generate high-quality images following text prompts [330, 342, 348] or even replicating styles and objects in given example images [123]. These technologies started to raise questions about human roles in art-making practices—whether these are good enough to replace people. For instance, if algorithms are good enough, why not prompt these algorithms to get the desired digital arts?

I argue that AI technologies can still extend human capability instead of replacing people. The key would be fitting high-quality generation results into the human workflow. For instance, in art commissions, even high-quality results might have aspects that the client or the artist does not like. If users can decompose these results into smaller elements, they would be able to let AI automate only a portion of the results while working on other things as they wish. It would allow users to keep their sense of ownership about the piece as they can work on core aspects by themselves. Ideally, with such support, users should be capable of introducing things that are different from what these algorithms tend to generate. I believe such an approach to be an interesting and important area for future work. Simply expecting artists to accept broad automation—even if the goal is to help them ‘prototype’ an artistic piece—will likely encounter resistance.

Whether generative artifacts can be leveraged to automate physical art-making is an interesting question. Certain types of physical art-making can involve AI generation with automation [155]. However, those art modalities where physical human skill is important would likely require more technological advancements (e.g., in robotics). Hence, technical requirements for automated physical art generation are high compared to digital art-making. Such developments are not impossible, but it will become critical to better understand the ‘fit’ of collaborative tools with generative capabilities within these mediums.

5.7.4 AI-CSTs in Unbalanced Collaborative Contexts

A few past research efforts have focused on using AIs in a collaborative art context [389]. However, these mostly looked into collaborative settings where the user roles were not necessarily different. My study expands knowledge of collaborative AI-CSTs by investigating settings with different user roles. First, due to different expertise and languages, *Artinter* has features that help users identify common ground about each other’s preferences, styles, and languages while serving to train the system. With AI functions that do not require art-making skills, we also observed that the tool could support non-experts to be more engaged in the process. With shared communication contexts and client’s extended engagement in the process, *Artinter* could allow users to more flexibly negotiate and balance the artist’s creative freedom and the client’s control, as mentioned in Section 5.5.3.5. Moreover, while AI functions can serve as a psychological safety net for non-experts, such a role was rarely mentioned by artists. One potential reason is the asymmetry in the expertise. As artists have more expertise than clients, they might have pressure to perform at an expert level, even with AI functions.

There can be many future work directions to support these unbalanced collaborative contexts. Within commission settings, I found that artists need to periodically update the client. Hence, there can be a tool that allow artists and clients to track the commission process after the initial meeting. For such a tool, an artist would be able to upload their intermediate work, and the client can check how the uploaded work meets agreed specifications. As clients might not have enough expertise to inspect and give specific feedback, the tool can have AI features to help with those (e.g., AI calculating which specifications the uploaded work satisfied). For artists, AI features could suggest some blueprints on how they can satisfy feedback from clients. With such a tool, we would also be able to investigate how AI technologies could support the end-to-end art commission process, not only the initial meeting. In general, we can design collaborative art-making tools with AI features to facilitate non-expert participation while easing the expert’s difficulties. Another interesting future work in unbalanced art-making collaboration would be in the contexts where people with different expertise work together to create a single artifact (e.g., a movie or a game).

5.8 Conclusion

I present *Artinter*, a collaborative AI-powered system that supports specifying art commissions through augmented boundary objects. *Artinter* helps artists and clients share boundary objects, verbal concepts, and artifact instances on a mood board and sketch interface. The system helps users minimize ambiguity in using boundary objects and expand the pool of artifact instances. To concretize boundary objects, *Artinter* allows users to ground them by defining verbal concepts with artifact instances through mood board interactions. To support expanding boundary objects, *Artinter* provides two AI-powered supports driven with slider interactions: 1) searching references with user-defined concepts, and 2) generating high-fidelity sketches by combining artworks or concepts. From two user studies, I found that mixing multiple input modalities in *Artinter* can support pairs of clients and artists in grounding their boundary objects while facilitating searching and generating references with learned concepts as steering handles. I also found that those learned concepts helped users iterate on search and generation to collect instances that can better explain their visions. Moreover, I identify other diverse usage patterns and limitations of *Artinter* that can guide the design of future AI tools for art commission communication support. In summary, this chapter demonstrates how the combination of the mood board and slider interactions can facilitate the steering of AI functions in visual art communication settings.

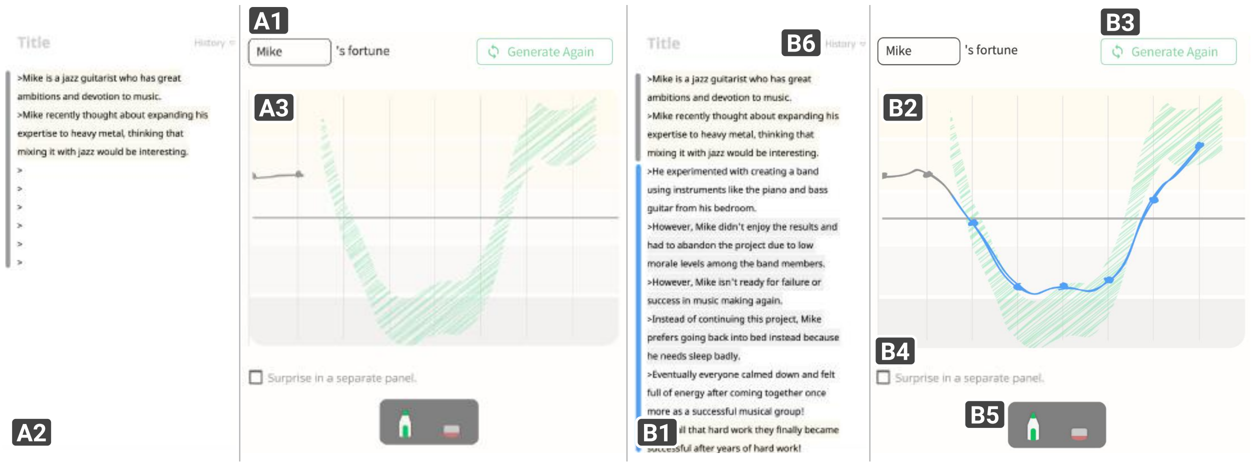
CHAPTER 6

TaleBrush: Steering of Story Generation with Visual Sketching

6.1 Introduction

Advances in pretrained generative language models, such as Open AI's GPT-3 [329], have enabled new kinds of human-AI story co-creation tools [48, 64, 69, 70, 169, 170]. In many of these tools, the story co-creation process is iterative [48, 69, 70, 148, 410]: first, a writer gives the initial story sentences, such as "Melissa fought a dragon." Then, using the writer's sentence as a prompt, the AI tool appends story sentences or phrases. For example, the tool can generate: "She used magic to create a barrier around her." The writer can modify the story directly, start from scratch, hoping for better results, or continue the iterative process by adding another sentence.

However, consider a case where the writer has some idea of how the story should unfold. They would like to build tension for Melissa by giving her some bad luck with the dragon and, only in the end, boost her fortune. Unfortunately, steering the text generation in this way is hard both from the algorithmic and interaction perspective. First, most controlled story generation algorithms, which can be guided with inputs like topical keywords, apply the same control to the whole generated story. However, as the writer wanted Melissa to experience bad luck first and then end up with a happy ending, they would need to specify different parameters for different parts of the story. Such granular sequence control is not yet possible in many controlled story generation algorithms. Second, even when the generation algorithm allows such granular sequence control, existing interfaces (e.g., text inputs or sliders) are not intuitive or useful for controlling or sensemaking in the generative process. For example, when there is a series of granular controls (e.g., sliders), specifying them can be cumbersome (e.g., inputting multiple numerical control values). Sensemaking, or understanding what the algorithm will do, or has done, is similarly difficult. To understand if the generated story follows the given parameters, writers have to compare each generated story part with each granular parameter. These frictions make iterative co-creation slow and effortful.



>Mike is a jazz guitarist who has great ambitions and devotion to music.
>Mike recently thought about expanding his expertise to heavy metal, thinking that mixing it with jazz would be interesting.
>
>
>
>
>
>

>Mike is a jazz guitarist who has great ambitions and devotion to music.
>Mike recently thought about expanding his expertise to heavy metal, thinking that mixing it with jazz would be interesting.
>He experimented with creating a band using instruments like the piano and bass guitar from his bedroom.
>However, Mike didn't enjoy the results and had to abandon the project due to low morale levels among the band members.
>However, Mike isn't ready for failure or success in music making again.
>Instead of continuing this project, Mike prefers going back into bed instead because he needs sleep badly.
>Eventually everyone calmed down and felt full of energy after coming together once more as a successful musical group!
[illegible]ll that hard work they finally became successful after years of hard work!

Figure 6.1: *TaleBrush* uses a line sketching interaction for intuitive control and sensemaking of story generation with GPT-based language models, closing the gap in the iterative co-creation process between humans and AI. In our prototype, the protagonist's 'fortune' is controllable. The writer first decides the protagonist's name (A1), writes a portion of a story (A2), and then sketches how the protagonist's fortune should change in the story (A3, green shaded area). The x position of the sketched line indicates the chronological position in the story (sentences are the units). The y position shows how good or bad the protagonist's fortune should be (higher, better). The width of the line indicates the possible variance in the fortune of the generated sentences. Given the line sketch, *TaleBrush* will generate story sentences (B1, indicated with blue) and visualize the result on the original sketch (B2, the blue line and dots). The writer can always directly edit the generated text. They can also iterate by generating new text for the same sketch (B3, clicking on 'Generating Again') or revising their sketch and generating new text. Additional options allow the writer to specify how 'surprising' the generation should be (B4) or using the eraser tool (B5) to erase their sketch. Where the line is erased, *TaleBrush* generates unconstrained sentences. A history dropdown (B6) allows the writer to browse previously generated sentences.

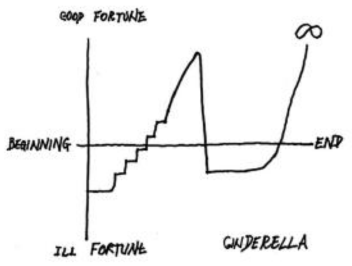


Figure 6.2: Kurt Vonnegut's sketched line drawing of *Cinderella's* fortune over the course of her story [408].

In this work, I propose that visual sketching interactions can facilitate granular sequence control and sensemaking in iterative story generation. Sequential properties of stories have frequently been visually expressed for intuitive understanding or planning [167, 192, 213, 250, 325, 333, 395, 396, 408]. For example, Kurt Vonnegut [408] drew out how a character's fortune changes in a simple time-series line drawing (Figure 6.2). This line drawing approach has inspired writers to adopt visual arcs to plan out their stories [85, 274, 334].

With this visualization as inspiration, I created *TaleBrush* (Figure 6.1). *TaleBrush* is focused on human and AI co-writing short story outlines by allowing writers to steer the generation according to the level of fortune the protagonist experiences [85, 333, 408]. Writers can specify the sequence of a character's fortune with a *simple* and *expressive* interaction (Figure 6.1A3). Using a single stroke (*simple*), the writer defines the fortune over the entire protagonist's 'time-series' (*expressive*). The sentences generated by *TaleBrush* are reflected back to the user as a line drawn on top of the original sketch (Figure 6.1B2). The lightweight and intentionally ambiguous nature of sketching [49, 84] manages the writer's expectation regarding the fidelity of steering, making uncertainties and errors in steerability acceptable. To realize this interaction, I implemented a technical architecture that achieves steerable story generation by training 1) learnable prompts that guide a language model to serve different story generation tasks [235] and 2) a control module that steers the language model to generate stories according to the given sketch [223].

I conducted a technical evaluation of *TaleBrush* and found that my approach can reliably steer generations that are still novel and tolerably coherent in flow and grammar. From a user study on 14 participants with varying expertise in story writing, I found that participants could use the sketching control to iteratively steer the generation to find inspiring stories that align with their intentions about the protagonist's fortune. Some participants also used line sketches as a plan when editing the generated texts. In our discussion section, I lay out the design elements for interactions that facilitate effective, efficient, and iterative human-AI co-creation. I also reflect on how visual sketching interactions can be extended to other story-writing attributes and domains while supporting reliable steerability even under the uncertainty of machine learning algorithms. This chapter introduces a way of using visual controls and visualizations to make iterative human-AI co-creation frictionless. This chapter is based on a published CHI2022 paper, “TaleBrush: Sketching Stories with Generative Pretrained Language Models” [66].

To summarize, this work contributes: 1) A line sketching interaction for steering and sensemaking of story generation with an abstract graphical representation of sequential attributes. 2) A GPT-based steerable language model architecture that generates story sentences with the sketching input on the protagonist’s fortune. 3) *TaleBrush*, a system that enables human-AI story co-creation with the protagonist’s fortune arc by combining line sketching interaction with GPT-based steerable language model. 4) Reflections on efficient steering interaction for iterative human-AI co-creation and how line sketching interaction can be expanded to other generative contexts.

6.2 Related Work

I review research on four main areas related to *TaleBrush*: 1) writing support tools, 2) story generation, 3) visual expressions of stories, and 4) sketching.

6.2.1 Writing Support Tools

Many tools support different aspects of writing. These range from spelling and grammar correction [231, 318], to thesauruses [130], to crowd-powered editors [32, 295]. As writing tasks and styles are often domain-specific, tools can be similarly specialized: email message phrasing [46], help requests in professional contexts [182], mental support [316], affectionate messaging [215], education [45], and journalism [266]. Of specific interest to us are tools for creative writing tasks. These run the gamut from suggesting metaphors [131] to helping with song lyrics [415]. Commercial tools such as Dramatica or Plottr are largely human-driven and help the author *structure* their plot or narrative [42, 109].

To *produce* content, some tools connect the author to other humans or support collective story writing by writing stories with the crowd. These systems can structure creative leadership [210] and decision-making [212], or even simulate the characters with crowds [173]. Newer tools have adopted novel machine learning (ML) and natural language processing (NLP) techniques [436]. Among those techniques, ML’s generative functions offer a powerful alternative approach. These systems append or suggest generated texts to the writer’s text using sophisticated language models [48, 69, 70, 358, 410]. In theory, these algorithms can act as writing support tools. However, limited controls and interactions—largely rephrasing prompts and contexts—constrain their applicability. My goal with *TaleBrush* is to leverage these language models but provide an alternative steering strategy using both text and abstract visual representations and interactions.

6.2.2 Machine Story Generation

Story generation is one of the grand challenges in artificial intelligence (AI) with multiple practical applications. These range from entertainment [340] and education [270, 340] to my target, writing support [48, 69]. Early approaches included the use of story block templates that could be put together sequentially [71]. With computational techniques, autonomous story generation became feasible. One technique is computational planning, where the computer does symbolic planning to accomplish a given goal [232, 279, 314, 339, 414]. Case-based reasoning techniques enable story generation by adapting stored stories to new contexts [132, 324, 338, 391, 401]. Alternative strategies involve character-based simulation, where characters and the world are simulated to unfold the story with the given facts and underlying beliefs [52, 255].

Advances in ML, largely in language models, have introduced more opportunities in story generation [15, 16, 153, 174, 185, 242, 244, 271, 394, 413]. Specifically, Transformer-based Language Models (LMs) [44, 329, 405] probabilistically sample continuing stories [105] or infilling sentences [15, 89, 180, 185, 242, 283, 303, 413] based on the given story context. While imperfect in generating coherent text, they hold promise for improved story generation.

Steerability is a key requirement for all these approaches. However, LM models, in particular, utilize structures that are much harder to inspect and explain. For the algorithms to be useful and usable, new steering mechanisms are needed [69]. The most basic, and perhaps obvious, steering approaches are driven by natural language prompts [93] (e.g., “Tell me a happy story”). To better guide LMs, researchers investigated ways to automatically learn these prompts [235, 238, 251, 363]. Other language-based approaches for steering involve specifying keywords to prime the generated text [104, 185, 390, 430]. Alternative approaches, like Intent-Guided Authoring (IGA), provide some degree of steering over output sentences [390]. The author can use a number of pre-specified tags (e.g., *cause* or *effect*) to describe the desired relationship of output to input. Researchers also introduced algorithms that generate stories while considering the character goals or abilities [5, 394]. Control codes, such as genre (e.g., is the story fantasy or science fiction?), can also provide some guidance. Technically, this approach has been enabled by training models with control codes [154, 206], or manipulating intermediate representation layers of LMs [53, 75]. Researchers have also introduced the approach of first acquiring keywords about the code, and generating the story based on them [221].

As LMs grow with more parameters, adopting these approaches became more challenging, as most LM parameters need to be tuned. To overcome this, researchers train smaller models that guide the large LMs with control codes [223]. While many approaches apply the steering to the whole story (e.g., 'the entire story should be science fiction'), a few approaches demonstrated how different parameters can be applied to different parts of the story [244, 332, 437]. However, they did not consider the interaction of the user steering in generation algorithms. Building on this prior work, I extend both the technical and interaction sides of steerable story generation. I designed my story generation model to be steerable with real-valued sequence inputs from line sketching interaction. There has been a relevant thread of work called visual story telling [30, 168, 169, 170, 179, 411], that writes stories about visual scenes (e.g., a photo of a dog running becomes "A dog was chasing down a rabbit"). In contrast, *TaleBrush* uses sketches of an *abstract* attribute of the story (e.g., the protagonist's fortune).

6.2.3 Visualizing and Visually Expressing Stories

To allow users to specify sequential attributes of the story and capture how texts are generated with steering, *TaleBrush* adopts abstract visual representations. Visual techniques have been used to show high-level attributes of documents [202] and stories (e.g., [39]). Digital humanities researchers have used visualizations to understand stories–a form of "distant reading" [192]. Through "distant reading", diverse story components can be visualized. These range from character interactions [33, 37, 195, 302, 420], event progression [167, 250, 395, 396], character emotions and sentiment [22, 33, 82, 161, 293, 333, 351]. As some stories can be non-linear in their telling, researchers have found ways of representing broader narrative structures [213, 359]. My goal is not simply to represent the story through the visualization but to allow for manipulation through that representation. However, not all visualization formats are naturally amenable to direct manipulation (beyond the standard interaction techniques [439]).

With *TaleBrush*, I focus on visual encodings that can both express high-level changes in the story's progression but also be manipulable. Kurt Vonnegut's story arcs fit these requirements. The arcs were used by Vonnegut to express canonical story types with diagrams that show how a protagonist's fortune changed in the story [408]. For example, in Figure 6.2, *Cinderella's fortune* is expressed with a curve that rises, drastically falls, and then rises again. Writers have since leveraged this type of visual expression to plan out story writing [85, 274, 334]. *TaleBrush* leverages this approach to allow for drawing a property, such as fortune, over time as a way of guiding the generative process.

6.2.4 Sketching Interactions

To visually express sequential attributes of stories, *TaleBrush* uses sketching interaction. Sketching is a flexible and lightweight way to convey the user's high-level intentions [99] with its roughness, uncertainty, and ambiguity [38, 144, 229]. Because of these features, various sketching tools have been built to embody these characteristics [144] or act as mechanisms to transform rough ideas into precise representations (e.g., visual renderings [229]). This interaction style has some similarity to *TaleBrush*, but without the direct mapping between objects (e.g., draw a cat to get a photorealistic cat). In the case of *TaleBrush*, I created a sketch in the abstract time-series space that will generate a corresponding story in the 'concrete' text space.

While sketching is most often used in a 'direct' way (e.g., drawing a sketch), it also has been used for constructing examples (e.g., for image [100, 176] or 3D model search [216, 305]). More relevant to my approach are tools for sketching time-series 'queries' for finding relevant items in time-series datasets. These can be used to find everything from fixed patterns [205] to constrained inputs [166, 346] to free sketches [97, 106, 268, 370, 416, 451]. However, most systems here assume the existence of matching datasets. In my case, the dataset does not yet exist. Thus, my approach does not sit cleanly in the regular taxonomy of visualization interaction [157, 439]. With *TaleBrush*, I leverage recent work on interactive visualization tools for data generation. However, these are largely focused on numerical approaches rather than text (e.g., generating a dataset that matches a sketched histogram [269]).

6.3 Sequence Steering Through Sketching

Though narratives may not be linear, stories are inherently formed in a sequence, progressing from the story beginning to the ending. Within that sequence, attributes of the story or characters (e.g., the protagonist's fortune) can change. My goal is to allow the user to use these attributes to guide the generated sequence. Specifically, I seek a lightweight mechanism to quickly specify multiple temporally-varying attributes. In this work, I consider line sketching interaction as the modality. There are a number of features of this interaction modality that make the approach appropriate. First, line sketching in time series visualization can be *expressive* enough to convey sequence and attribute information [3, 290]. For example, in Figure 6.2, the sequence can be shown on the x-axis while expressing the protagonist's fortune on the y-axis. Second, line sketches are *simple*. In Figure 6.2, the protagonist's fortune is expressed just in a single stroke of a line. Though I could build a similar time series through sliders, text entry of coordinates into a spreadsheet or clicking on points, a drawn line in a time-series is lightweight and intuitively understood. This design is well suited for continuous values (e.g., the fortune level). However, one could imagine 'sketching' discrete/categorical values as well by drawing disconnected horizontal lines. By default, a simple swipe can allow the author to indicate the attribute, for how long it should last, and where in the story it should exist.

While **sequence** can be expressed in various ways, I choose to use the familiar linear x position (with attribute values on the y). Linear positions have been most frequently used to visualize

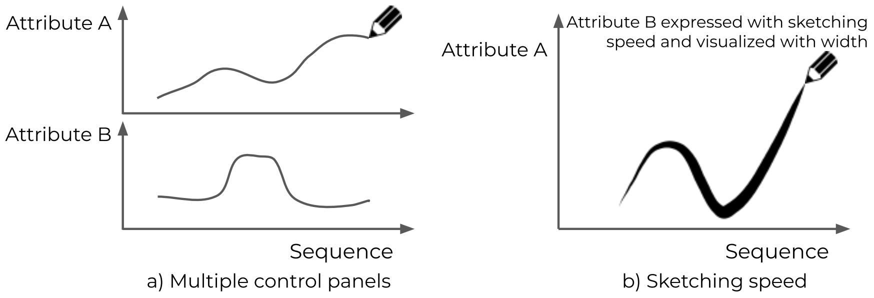

Attribute A
Attribute B
Sequence
a) Multiple control panels
Attribute A
Attribute B expressed with sketching speed and visualized with width
Sequence
b) Sketching speed

Figure 6.3: More than one story attribute can be expressed with a) multiple steering panels or b) the use of sketching speed.

sequence and it can promote accurate perceptual judgment of time [3, 41]. Rotated (e.g., clock) or spiral-format temporal representations are also possible [417] but are best suited for representing cycles or seasonality in temporal data—something uncommon in stories.

In some cases, we may want to express **multiple story attributes**. For example, the user might want to specify two characters' fortunes. In this situation, the multiple time series can either be sketched within the same $x - y$ space (assuming the scales are the same) or drawn in multiple spaces (e.g., Figure 6.3a). To draw two attributes (along with time) in a single plot is possible but requires additional encodings. For example, a connected scatterplot (see [152]) draws the values for the two attributes on the x and y axes respectively with time encoded using marks on the lines themselves (e.g., arrows or graduated thickness) that indicates sequence. One could also imagine using a feature of the sketching interaction (e.g., pressure or the time it takes to draw a segment) to encode values. For example, in Figure 6.3b, time is displayed on the x-axis, the value for attribute A on the y-axis and the time used to draw the stroke (or pressure) to encode the B attribute (visually these can be encoded using stroke size or color). Drawing speed is naturally combined with stroked lines and allows for a semantic connection between fast-slow drawing speeds, and high-low attribute values. Additionally, users may have ambiguous perceptions of speed and time [323]. This might further the idea that the generated story will only roughly follow the input. Ultimately, this encoding is likely inappropriate. It still limits us to two attributes and is likely complex and discordant as two similar variables (i.e., attributes A and B) are created and visualized in two different ways.

A potentially better use of the drawing speed (or pen pressure) is to convey ambiguity and error-tolerance. In most cases, a generative algorithm will not be able to generate a sentence that precisely matches the attribute value. This is due to two factors. First, is the randomness in the generation (i.e., we may not be able to generate a sentence with a character's fortune at exactly 0.58). Second, is the inherent noise in any classifier that judges attribute values based on the text. By using 'sketchiness,' we are already conveying some level of ambiguity. However, it would be ideal to allow the end-user to be able to indicate how accurately they want the generated sentence to map to their sketch. To allow this specification, sketch speed or pen pressure may useful. Both actions have a natural mapping to 'dropping more ink' in natural pen interactions. The size of the sketched line at a particular time indicates the desired bounds–smaller, indicating more constrained. Notably, the specific semantics of slower motion can be switched. Drawing slower can be made to produce 'more ink' and therefore higher tolerance. Conversely, people often draw more slowly when they want an 'accurate' shape. Thus, a slower draw speed can semantically map to a more 'accurate' (i.e., tighter/thinner) line being drawn.

Though there is a fairly broad design space for sketched input, the combination of interactive steering, representation effectiveness, and flexibility seem best addressed through simpler representations: standard time-series for a single attribute and additional panels for multiple attributes. However, stroke speed or pressure, represented as line thickness or color, allows us to encode additional information such as the user's ambiguous intentions and error-tolerance.

6.4 *TaleBrush*: Interface

In this section, I motivate our selection of steerable story attributes and explain interactions in *TaleBrush*'s interface.

6.4.1 Iterative Co-creation With Line Sketching

TaleBrush is designed to help writers co-create a short storyline with AI through sketching interactions. *TaleBrush* supports the planning stage of the writing process [111, 142] by giving ideas that can be novel to writers. I chose to support this specific stage, as writers would likely accept and benefit from diverging generations. After a story is generated, writers can freely edit it or even try the generation again to see another story from *TaleBrush*. *TaleBrush* can support writers who would use generated sentences as writing prompts, or even potentially help them overcome writer's block. For novice writers, *TaleBrush* would demonstrate a story writing that would allow them to easily jump into writing.

In *TaleBrush*, writers iterate on the story generation to reach the desired state. Through interactive steering, the *TaleBrush*'s goal is to allow writers to reach this state with a small number of iterations. As introduced in Section 6.3, sketching (green shaded area in Figure 6.1A3 and B2) allow for *simple* and *expressive* steering interaction. *TaleBrush* visualizes generated stories in the same abstract visualization (blue line in Figure 6.1B2). Through this, writers can more easily understand the relation of their input to the generated content. I believe that the intuitiveness of these natural steering and visualizations will ultimately allow writers to more easily iterate on the story generation.

In my initial implementation, *TaleBrush* specifically focuses on two steerable attributes. The first is **the level of the protagonist's fortune**. This is based on the existing practice of writers expressing and planning stories based on character or emotional arcs [85, 418] or more generally on a character's fortune [408]. *TaleBrush* is designed to allow the steering of the protagonist's fortune in chronological order. Non-linear narratives, such as flashbacks, are not currently supported. The user can also specify how tightly *TaleBrush* would follow these given fortune parameters at the cost of generation time. The second attribute is **the level of surprise** in the generation, or how unexpected the generation should be.

6.4.2 Interface

6.4.2.1 Text Editor and Canvas

TaleBrush has a text editor (Figure 6.1A2 and B1) and a canvas for drawing and visualizing the story arc (Figure 6.1A3 and B2). The text editor has multiple 'bullet points', each of which stands for a single sequence step, most often a sentence. The writer can add their own sequence sentences or ask *TaleBrush* to generate them. As with a standard editor, the writer can add new sequence sentences (simply by hitting enter), or remove them by deleting the bullet point line. *TaleBrush* will visualize each bullet point line as a dot in the canvas and sequential dots will be graphically connected with lines. Thus, there is a correspondence between the bullet point order and the x position of those sentences in the visualization. Missing sentences (i.e., blank bullet points) are reflected with a visual gap. The protagonist's fortune in a given sentence is determined by an ML recognition algorithm.

6.4.2.2 Generate By Sketching

After the writer sketches a character's fortune (as described in Section 6.3), *TaleBrush* generates text for those sentences for "sketched" sentence slots. For example, the writer can enter one or two sentences, for which the fortune is calculated and displayed. They can then sketch the rest of the sequence. *TaleBrush* also supports 'infilling' the story. For example, if the writer enters a sentence at the start and end of the sentence list, this will appear as a broken line in the visualization. The writer can sketch how they want the story to behave in this empty part and *TaleBrush* will create suitable sentences. The writer can also draw 'broken' lines by erasing part of the sketched line. In this mode, *TaleBrush* will generate a suitable sentence but without any specific fortune 'target.' This allows writers to explore stories even when they do not have a specific idea on what the character's fortune should do. Once the sketch is drawn, the writer can perform generation again with the same parameters by clicking the "Generate Again" button (Figure 6.1B3). They can also redraw a line only on a part of the sequence to see other generated sentences on the redrawn part. In all cases, the writer also can revise generated sentences directly in the text editor.

Sketching Speed for Generation Steering Tightness As the steerable generation has uncertainty in how the steering would apply, with the sketching speed, the writer can control their ambiguous intentions by setting how closely generated sentences should follow the given sketch. With slow sketching, *TaleBrush* tries more generations to find the sentence that better matches the given fortune parameter. Through iterative prototyping, I found that the *fast* $\rightarrow$ *less accurate* and *slow* $\rightarrow$ *more accurate* mappings were easier to understand. The drawn sketch can be perceived as an 'envelope' representing the bounds in which the generated sentences would be more likely to fall (Figure 6.1A3 and B2). We detail how the specific width is determined in Section 6.6. The width of the sketched line gradually changes visually (i.e., it is not disjoint). This is more appealing visually. However, the actual envelope is determined using sentence 'units.' Subtle vertical lines indicated where these transitions are, thus allowing the user to vary their drawing speed in each segment.

Surprise Level Control As I describe below, the used language also allowed users to change parameters that roughly translated to how 'surprising' a sentence would be in the following prior sentences. For example, with the beginning sentence of "Melissa fought a dragon", the low surprise may generate "The dragon was a big, green, scaly beast", but the high surprise may result in "Melissa killed a giant robot with a robot dragon". To experiment with this feature, I added a "surprise in a separate panel". This would open up another time series canvas where the writer could draw the desired level of surprise for each sentence segment (see Figure 6.4).

6.4.2.3 Multi-Story Management

As the writer generates multiple stories, *TaleBrush* stores each. Users can compare and choose among them (Figure 6.5). A dropdown menu lists the generated stories, which will be displayed on the canvas with low opacity. As the writer moves the mouse over the list, the hovered story is shown in the text box and highlighted on the canvas. The writer can select one of these to 'roll back' to a past generation.

6.4.3 Implementation

The interface for *TaleBrush* is implemented as a web application, using HTML, CSS, JavaScript, and React. The language model operations for text generation are implemented in a back-end server. A Flask-based REST API is used to connect the front– and back–end sub-systems.

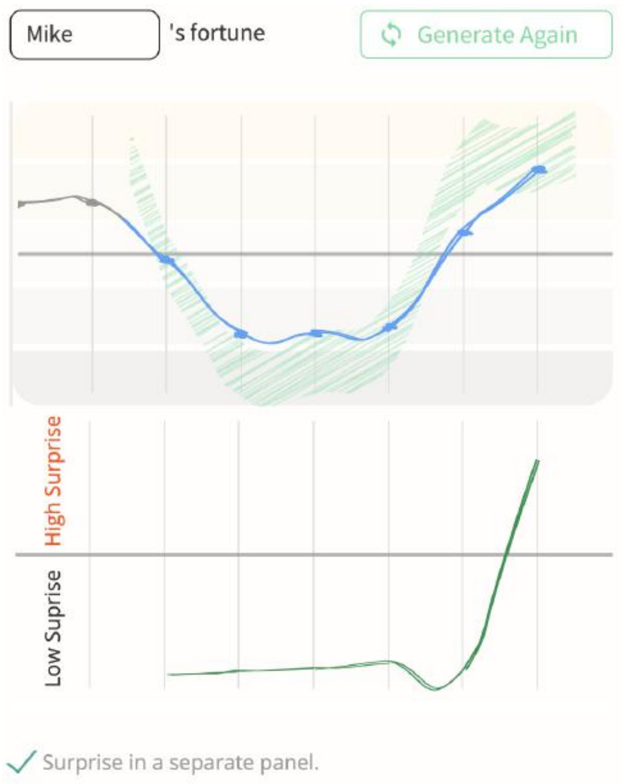


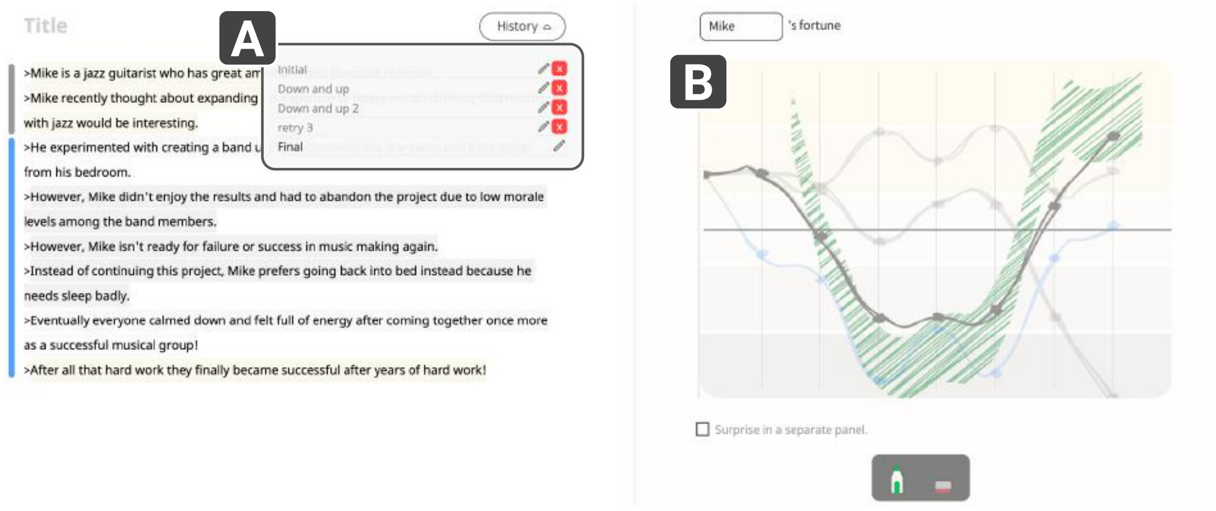
Figure 6.4: *TaleBrush* allows writers to specify the level of 'surprise' for each generated sentence. A separate canvas can be opened where the writer can sketch a thin green line indicating the desired level of surprise. Here, we see a fortune arc for Mike (top) starting high, going low, and ending high. A corresponding surprise sketch (bottom) indicates that the author wanted the first part of the story to be more controlled but the end should be very surprising.

6.5 *TaleBrush*: Technical Details

There are two principle technical components to *TaleBrush*: sentence **recognition**, which determines a fortune level given an input sentence; and sentence **generation**, which creates new sentences. I begin by explaining our annotated training data, and then cover each of recognition and generation modules.

6.5.1 Collecting Fortune of the Protagonist

I adopted crowdsourcing to create an annotated fortune dataset, as there is no readily available data of this type. I define the protagonist's fortune as *level of 'goodness' or 'badness' of the fortune experienced by the protagonist, as perceived by the readers*. This construct is naturally affected by the character's status, emotional and physical well-being, or spatial, mental, and emotional proximity to the character's goal [85]. While line sketching interaction would require fortune



>Mike is a jazz guitarist who has great am

>Mike recently thought about expanding

with jazz would be interesting.

>He experimented with creating a band u

from his bedroom.

>However, Mike didn't enjoy the results and had to abandon the project due to low morale levels among the band members.

>However, Mike isn't ready for failure or success in music making again.

>Instead of continuing this project, Mike prefers going back into bed instead because he needs sleep badly.

>Eventually everyone calmed down and felt full of energy after coming together once more as a successful musical group!

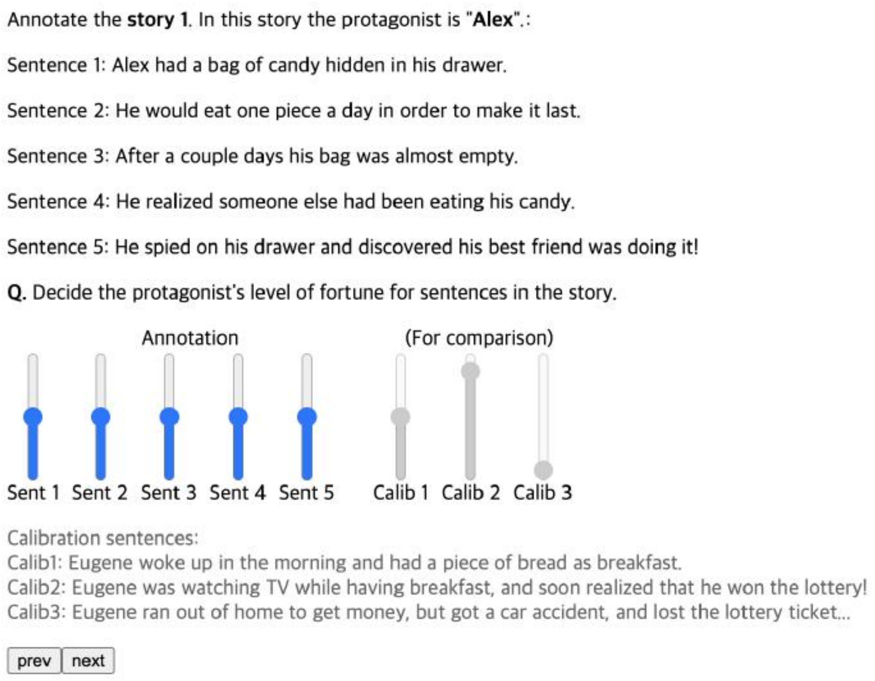
>After all that hard work they finally became successful after years of hard work!

Figure 6.5: *TaleBrush* stores generated stories in a list (A) and allows writers to compare them on the canvas by visualizing the fortune arcs of all stored stories in low opacity (B). The writer can hover over stored stories in the list to see their text and story arc visualization.

annotations to be real values, due to the subjectivity in perceiving this concept, directly collecting real-valued annotations can result in a high variance. As people can make comparisons reliably on subjective annotations [83, 181], I took the approach similar to previous work's [83]: turning crowdsourced comparisons into real-valued annotations.

I collected fortune annotations on a subset of the ROCStories dataset [288]. Items in this dataset are short five-sentence stories. I randomly sampled 2200 stories, using 2000 for training and 200 as a test set. I assigned three annotators for each story. During data collection, I explicitly specified the protagonist of the story. Annotators were asked to specify the level of the protagonist's fortune for each sentence using a continuous slider (Figure 6.6). While this annotation approach is mainly designed to collect real-valued annotations, the interface also enables annotators to encode comparisons between sentences. Workers were not explicitly asked to compare sentences between different stories. However, they were asked to annotate using the "calibrating sentences" as benchmarks. I crafted these calibrating sentences from a three-sentence story to ensure a unique, single-sentence example for three fortune 'types': good, bad, and neutral. Annotators were asked to label these calibrating sentences at the start of the task and could see them as they worked (see Figure 6.6).

To ensure high annotation quality, I included one gold standard question with sentences of obvious good and bad fortunes. I filtered out workers who annotated such questions in opposite fortune directions. I recruited annotators from Amazon Mechanical Turk, who are in the US and had 97% acceptance rate with 1000 or more accepted tasks. Each worker was asked to annotate 11 stories including the calibrating story and was paid \$2.50 (about \$10/hr payment rate).


Annotate the **story 1**. In this story the protagonist is "**Alex**".:

Sentence 1: Alex had a bag of candy hidden in his drawer.

Sentence 2: He would eat one piece a day in order to make it last.

Sentence 3: After a couple days his bag was almost empty.

Sentence 4: He realized someone else had been eating his candy.

Sentence 5: He spied on his drawer and discovered his best friend was doing it!

Q. Decide the protagonist's level of fortune for sentences in the story.

Calibration sentences:
Calib1: Eugene woke up in the morning and had a piece of bread as breakfast.
Calib2: Eugene was watching TV while having breakfast, and soon realized that he won the lottery!
Calib3: Eugene ran out of home to get money, but got a car accident, and lost the lottery ticket...

Figure 6.6: Interface for annotating the protagonist's fortune level. The annotator can make comparisons between sentences in the given story. Calibration sentences help for making comparisons between stories.

To learn the reliability of annotations, I analyzed agreements between pairs of annotators. I measured Spearman's ρ in ranking five sentences in a story according to the fortune level (i.e., with perfect agreement $\rho = 1$ or -1 with perfect disagreement). The average and median of ρ values between pairs were 0.54 and 0.7, respectively. Though clearly imperfect, annotators did display agreement in their annotations.

To turn comparisons into real-valued labels, I used the TrueSkill algorithm [83, 158, 280]. The algorithm can calculate the real-valued fortune score of each sentence out of all comparisons from our crowdsourced annotations. While this algorithm is originally designed to infer rankings of game players out of all individual game results, like who won or lost, I adapted it to get rankings of fortunes out of all comparison annotations. With this algorithm, sentences frequently annotated to have a higher fortune than other sentences would have a high fortune ranking. I normalized this ranking information to have a uniformly distributed dataset on the scale of 0 to 1 and used the resulting annotations.

6.5.2 Technical Architecture

A high level architecture for the recognition and generation elements of *TaleBrush* is depicted in Figure 6.7.

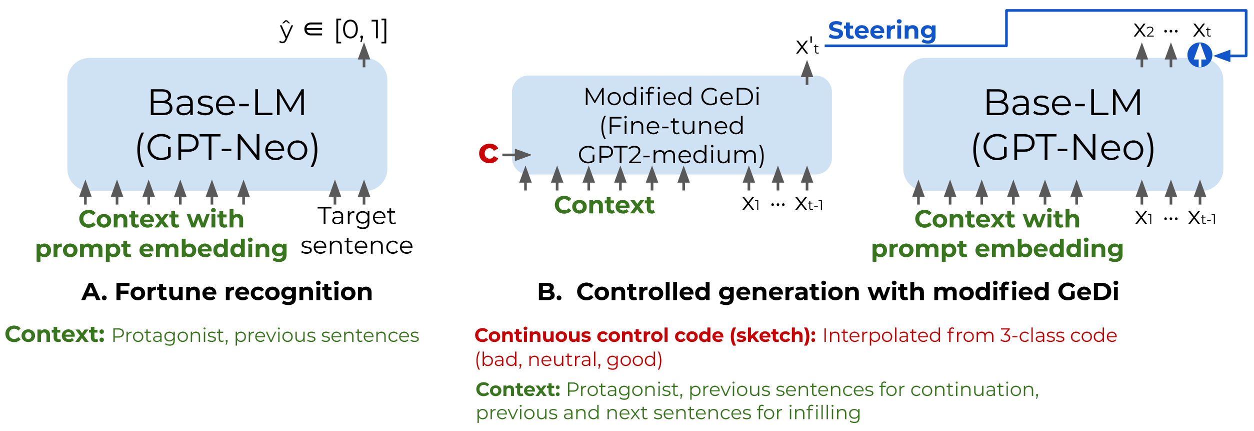
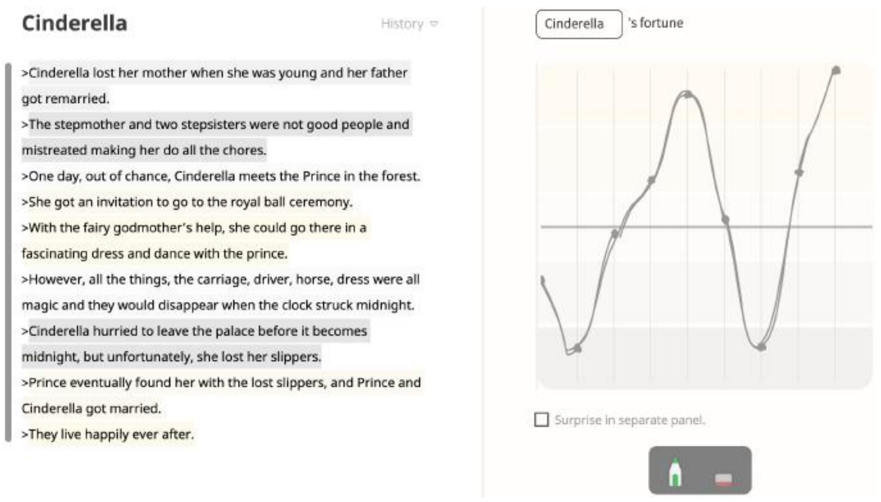

ŷ ∈ [0, 1]
Base-LM (GPT-Neo)
Context with prompt embedding
Target sentence
A. Fortune recognition
Context: Protagonist, previous sentences
Steering
X'ₜ
Modified GeDi (Fine-tuned GPT2-medium)
C
Context
X₁ ··· Xₜ₋₁
X₂ ··· Xₜ
Base-LM (GPT-Neo)
Context with prompt embedding
X₁ ··· Xₜ₋₁
B. Controlled generation with modified GeDi
Continuous control code (sketch): Interpolated from 3-class code (bad, neutral, good)
Context: Protagonist, previous sentences for continuation, previous and next sentences for infilling

Figure 6.7: The technical architecture of *TaleBrush*. To serve recognition and generation, the prompts of a Base-LM are tuned. These prompts are used interchangeably for a single LM. For fortune recognition, context, prompt embeddings, and target sentence are given as inputs and the fortune level is predicted as a real value between 0 and 1. For steerable generation with the fortune sketch, a modified version of GeDi is used. GeDi is a fine-tuned model that steers the generation according to real-valued fortune input, or continuous control code. Context and prompt embeddings are also used as input. For a sentence to be generated, the y-position of a point in a sketch is used as a continuous control code. This continuous control code has a value between 0 and 1 and its input embedding is interpolated with three class codes (bad for 0, neutral for 0.5, and good for 1) before being input into the modified GeDi. Base-LM used GPT-Neo while the modified GeDi used GPT2-medium.

6.5.2.1 Recognition Module

TaleBrush recognizes the protagonist's fortune to map sentences in the visualization. To match the training dataset for the recognition and steerable generation, I trained a recognition model. Specifically, I use prompting approaches of GPT-based models [44, 329]. For instance, to recognize the protagonist's fortune, to the model, we can input the prompt "Estimate fortune:" with the subject sentence appended to it. However, hand-designing optimal prompts is known to be challenging [235, 251]. Hence, I used the soft prompt approach [235, 238, 251], which learns these prompts on the continuous input embedding space. These soft prompts replace "discrete" natural language prompts. In other words, I can replace "Estimate fortune:" with several learnable soft prompt embeddings. I took the soft prompt tuning approach [235] instead of fine-tuning the whole parameters of the LM, as trained soft prompts require less memory (about $\times 10^5$ less size than the whole parameters). Fine-tuning multiple whole-parameter models for multiple tasks (e.g., generation), would require significantly more computation resources.

I trained the soft prompts to minimize the regression loss on the protagonist's fortune label of an estimated sentence. As a language model, I used a GPT-Neo [35], which is a GPT-based autoregressive model [44, 329] with 2.7 billion parameters. I included the context sentences previous


Cinderella

>Cinderella lost her mother when she was young and her father got remarried.
>The stepmother and two stepsisters were not good people and mistreated making her do all the chores.
>One day, out of chance, Cinderella meets the Prince in the forest.
>She got an invitation to go to the royal ball ceremony.
>With the fairy godmother's help, she could go there in a fascinating dress and dance with the prince.
>However, all the things, the carriage, driver, horse, dress were all magic and they would disappear when the clock struck midnight.
>Cinderella hurried to leave the palace before it becomes midnight, but unfortunately, she lost her slippers.
>Prince eventually found her with the lost slippers, and Prince and Cinderella got married.
>They live happily ever after.

Figure 6.8: How the protagonist's fortune is recognized in the summary of *Cinderella*. The visualized recognition results are similar to Vonnegut's original (Figure 6.2).

to the target sentence in the input data, as they can have critical information. This previous context includes, at maximum, three sentences. I picked three as it had the smallest loss in the test dataset. I also included the protagonist's name in the context. The input and output labels are formatted as follows:

- Input: [P^1_{prev}], ..., [P^n_{prev}], (previous context), [P^1_{prot}], ..., [P^n_{prot}], (protagonist), [P^1_{tar}], ..., [P^n_{tar}], (target sentence)
- Label: $y \in [0, 1]$

[P^i_r] stands for the placeholder for soft prompt embeddings, with r and i indicating the role and the index of soft prompts, respectively. For each prompt role, I chose three prompt tokens ($n = 3$) which fit with that of initialization prompts I used ('Previous Context:' for P_{prev}, 'Protagonist:' for P_{prot}, and 'Sentence:' for P_{tar}). With trained prompts, I could recognize the protagonist's fortune, as demonstrated on *Cinderella* in Figure 6.8.

6.5.2.2 Generation Module

The goal is for *TaleBrush*'s generation to be steerable with a sequence of fortune values. For this, I modified the approach used in GeDi [223], which uses a fine-tuned smaller model to guide a bigger language model with a steering input. In order to explain the details of our GeDi variant, I first describe how generation can be done with LMs. Then, I describe our base LM that serves story generation tasks and GeDi model that steers stories with the control code.

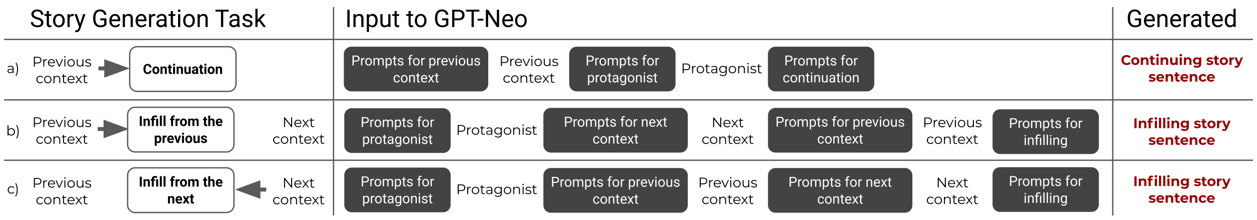


Figure 6.9: *TaleBrush* performs three story generation tasks: a) continuation, b) infill from previous, and c) infill from next. For different generation tasks, how contexts and prompts are structured for inputs differs. In *Input to GPT-Neo*, dark boxes indicate continuous prompts. In *Generated*, red text stand for generated sentences.

Generation with Language Model When an auto-regressive LM receives a sequence of tokens ($x_{1:T} = \{x_1, \ldots, x_T\}$), it calculates the probabilities for the next tokens with a chain rule ($P_\theta(x_{1:T}) = \prod_{t=1}^{T} P_\theta(x_t|x_{<t})$). With this probability, we can sample out the next likely token. By inputting the sampled token back into the LM, we can also calculate the probability for the token that comes after the lastly sampled token.

Base Language Model for Story Generation As a base LM for language generation, *TaleBrush* uses the same GPT-Neo the recognition module uses. However, different soft prompt embeddings are trained for the generation tasks. The benefit is that one large pre-trained LM can be utilized for different tasks just by replacing prompts and contexts according to the task purpose. I trained soft prompts for story continuation and infilling. For continuation (Figure 6.9a), the previous context and protagonist are fed into the language model as input and the model generates the continuing sentences. I appended the character information as models without character information tend to introduce random characters. I placed the character information right before the text to be generated, to make it more likely that character information is considered during the continuing generation. Accordingly, the input data is formulated as the following:

- Continuation: $[P^1_{prev}]$, ..., $[P^n_{prev}]$, (previous context), $[P^1_{prot}]$, ..., $[P^n_{prot}]$, (protagonist), $[P^1_{cont}]$, ..., $[P^n_{cont}]$

Similar to recognition prompts, I chose the number of prompts that matches the initializing prompt ($n = 3$ with 'Beginning Story:' for P_{prev}, 'Story Character:' for P_{prot}, and 'Continue Story:'for P_{cont}).

For infilling (Figure 6.9b and c), I included both the previous and next contexts and the protagonist as the model input. I trained two types of prompts for infilling: one for filling after the previous context, and the other for filling backward from the next context. When running infilling generation, these two prompts are used in turn, switching from one to the other for every generated infilling sentence. For infilling from the previous, I placed the previous context right before the text to be generated, as the generated text will be appended right next to the previous context. With a similar rationale, for infilling from next, I placed the next context right before the text to be generated. I placed protagonist information at the front-most prompt, as infilling tend to digress less with the protagonist compared to continuation. This likely occurs as the protagonist is mentioned in both the 'previous' and 'next' contexts. The input data for infilling was formulated as follows:

- Infilling from the previous: [P^1_{prot}], ..., [P^n_{prot}], (protagonist), [P^1_{next}], ..., [P^n_{next}], (next context), [P^1_{prev}], ..., [P^n_{prev}], (previous context) [P^1_{infill}], ..., [P^n_{infill}]
- Infilling from the next: [P^1_{prot}], ..., [P^n_{prot}], (protagonist), [P^1_{prev}], ..., [P^n_{prev}], (previous context), [P^1_{next}], ..., [P^n_{next}], (next context) [P^1_{infill}], ..., [P^n_{infill}]

I chose the number of prompts according to that of the initializing prompt. ($n = 3$ with 'Story Character:' for P_{prot}, 'Beginning Story:' for P_{prev}, and 'Ending Story:' for P_{next}, and $n = 4$ with 'Infilled Story:' for P_{infill}).

Steerable Generation with GeDi *TaleBrush* adopts the GeDi approach. This model guides an LM model (base LM) with a smaller Class-conditional LM (CC-LM) that is fine-tuned to generate using a user-specified control code (e.g., generate a happy story with the control code of "good"). Similar to standard LMs, CC-LM calculates the probability of the next token given a sequence of tokens. However, it also considers a specific *control code* (c) given to the CC-LM ($P_\theta(x_{1:T}|c) = \prod_{t=1}^{T} P_\theta(x_t|x_{<t,c})$). With CC-LM's probabilities for different codes and the Bayes rule, we also can calculate the probability of each sequence of tokens being classified into a control code:

$$P_\theta(c|x_{1:t}) = \frac{P(c)P_\theta(x_{1:t}|c)^{\alpha/t}}{\Sigma_{c' \in \{c,\bar{c}\}} P(c')P_\theta(x_{1:t}|c')^{\alpha/t}}, \tag{6.1}$$

where α is a learnable scale parameter and $\bar{c}$ is codes other than the selected one. This probability can be combined with that of base LMs, to decide which token would lead to the coherent story that follows the control code:

$$P_\omega(x_t|x_{<t}, c) \propto P_{LM}(x_t|x_{<t})P_\theta(c|x_t, x_{<t})^\omega \tag{6.2}$$

Note that ω serves as a parameter that decides how much steering will be done with GeDi (higher $\rightarrow$ more steering). GeDi is faster than other steering approaches that directly manipulate weights [75] or that filter from many generated texts. Thus, it is better suited for interactive applications.

Modified GeDi for Continuous-Value Steering I modified GeDi to steer generation with continuous-valued control codes. To enable continuous steering between good and bad fortunes, I

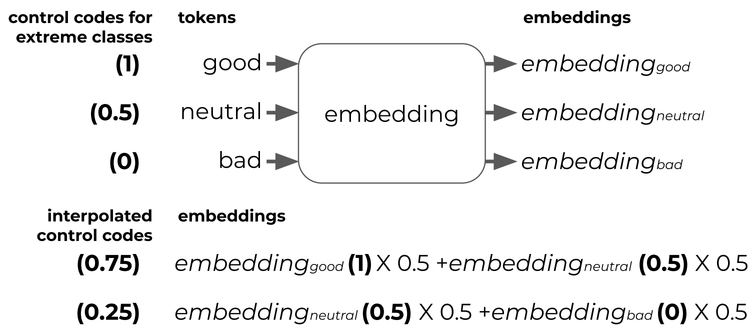

control codes for extreme classes
tokens
embeddings
(1)
good
(0.5)
neutral
(0)
bad
embedding
embedding_good
embedding_neutral
embedding_bad
interpolated control codes
embeddings
(0.75)
embedding_good (1) X 0.5 + embedding_neutral (0.5) X 0.5
(0.25)
embedding_neutral (0.5) X 0.5 + embedding_bad (0) X 0.5

Figure 6.10: An approach to turn continuous control codes into embeddings that can be input to CC-LM. For three extreme classes (1, 0.5, 0), we use embeddings of the tokens, *good*, *neutral*, and *bad*, respectively. For other control codes, I interpolated the embeddings of these extreme classes.

interpolated between extreme fortune classes. I first trained CC-LM with three topic classes, each mapping to extremely *bad* fortune, *neutral* fortune, and extremely *good* fortune. When training the CC-LM with these three classes, I took the multi-class topic steering approach introduced in GeDi [223]. With this approach, the given "main" codes are *true* and *false*, while three classes of *bad*, *neutral*, and *good* are given as the secondary codes. When given *true*, CC-LM trains to generate texts according to the given secondary code, and with *false*, CC-LM would not follow the secondary code. I adopted this multi-class topic steering approach as it has benefit in computation time, with fewer "main" control codes to be computed to get the probability from Equation 6.1. I also included the protagonist's name at the beginning of the input as contexts. A part of training dataset might look like:

- true good Protagonist: Mark # Mark was happy to pass the exam that he wanted to pass.
- false bad Protagonist: Mark # Mark was happy to pass the exam that he wanted to pass.

For the training data, I used only some of the annotations from my collection: those in the top, middle, and bottom 10% percentile. That is, I mapped ranges of values to three discrete classes to assure I have enough training data. To train this CC-LM, I fine-tuned the GPT2-medium model. After the CC-LM is trained, when an arbitrary value between 0 and 1 is used as a code, I interpolate embeddings between three discrete classes considering each of them as having 0, 0.5, and 1 (Figure 6.10). Then, I used the interpolated embedding as the input embedding. For example, when the control code is 0.25 (halfway between the bad and neutral fortunes), I calculated the weighted sum of the the learned *bad* and *neutral* embeddings with the weight of 0.5 each.

While GeDi steers the generation with the protagonist's fortune, it can potentially make errors, producing sentences outside of the desired fortune level. To handle these errors, I adopted the approach of regenerating sentences if they were significantly out of the range of the specified code.

Specifically, *TaleBrush*'s recognition module measures the generated sentence's level of fortune, and if it disagrees with the steering input value, *TaleBrush* tries to generate again. I set the threshold of error for regeneration as 0.2 (i.e., $|y - \hat{y}| > 0.2$). *TaleBrush* tries to regenerate the text 1-2 more times. Again, this is based on the desired 'closeness' of the generated fortune to the writer's intent (as expressed by the speed of the writer's drawn stroke). After the maximum number of regeneration has been tried, *TaleBrush* picks the generated sentence that has minimal fortune level error with the given steering input value. The surprise level control value is used as the temperature of the softmax function of the base language model. To implement our architecture, I used models from Huggingface[1].

6.6 Technical Evaluation

I conducted technical evaluations on the recognition and generation modules. For recognition, I measured how accurately a character's fortune is recognized in a sentence. For the generation module, I focused on how steerability impacts other qualities of the story. Steerability is known to have trade-offs with some story qualities like coherence [244]. Ideally, my control approach would allow the writer to steer the generation without hurting other metrics. I conducted three evaluations: 1) recognition performance, 2) an automated evaluation on generation, to find the parameter that can achieve the 'sweet spot' between steerability and coherence, and 3) human evaluation on generation, to show that my approach allows steerability while maintaining other story qualities.

6.6.1 Fortune Recognition Evaluation

To evaluate the recognition module, I focused on recognition error, the difference between the recognized fortune value and the gold standard labels. This was done on the 200 annotated examples reserved for testing. The mean and median errors were 0.179 and 0.159 respectively on a 0 to 1 scale. I considered this result to be reliable. Concepts similar to fortune, such as sentiment, are usually measured through an ordinal five-level scale. Human raters can easily discern concepts with this number of levels [375]. If we view the five levels on a 0 to 1 scale, with the assumption of the uniform interval between levels (which is widely adopted in ML [375, 442]), then the gap between levels will be 0.25. Our average error (.179) is below this 0.25 gap so most differences would not be easily discernible.

[1]https://huggingface.co/

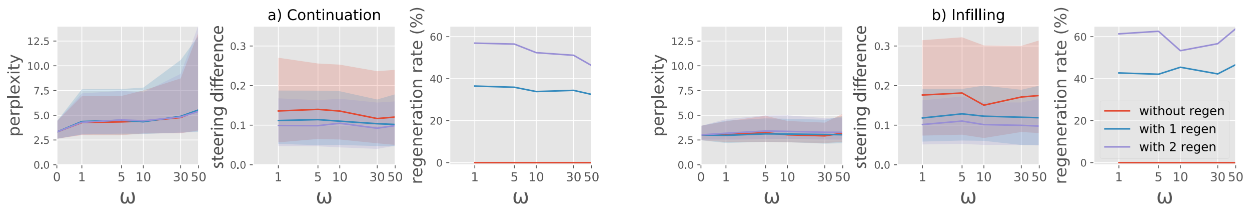


Figure 6.11: Automated evaluation results on steered story a) continuation and b) infilling. Median of perplexity and steering difference (between given steering input value and output fortune) are plotted for the range between 25^{th} and 75^{th} percentile. In the regeneration condition, we measured the rate of sentences that were regenerated to minimize steering difference. The x-axis is presented in log-scale.

6.6.2 Automated Generation Evaluation

I conducted the automated evaluation to identify the parameter that balances coherence and steerability. Here, the parameter of interest is ω: how intensely the steering is applied. Past work has demonstrated that a higher ω value leads to better steerability, but worse coherence [244]. As I modified the GeDi approach to our context, I reinvestigated how ω impacts coherence and steerability.

For coherence, I calculated perplexity, which is a measure of how well a probability model predicts to generate a sequence of text tokens [43]. To evaluate the steerability, I measured the steering difference: the difference between an input fortune value and the estimated fortune level of the text generated. To estimate the fortune level of generated text, I used our fortune recognition module. I measured these two metrics with our 200 test data points. For the continuation task, I considered the first sentence as the initial context and generated four more sentences. For infilling, I considered the first and the last sentences as the initial context and generated middle three more sentences. For the steered generation, I gave fortune annotations in our test data as steering input values and measured perplexity and steering difference, while varying ω value. Recall that I allow for 'regenerating' the text up to two times to improve the generated text's fit to the desired fortune (selecting the best of the generated options). I also simulated this regeneration up to two times to determine the impact of this on our metrics.

With steered continuation (Figure 6.11a), the perplexity tends to increase with the increase of ω. The increases in median and 25^{th} percentile were relatively stable compared to that of 75^{th} percentile. The steering difference tended to decrease very slightly with the increase of ω. The regeneration approach tends to decrease not only the median of steering differences but also the 75^{th} percentile of steering difference by a large amount. Regenerating once (two stories to pick from) resulted in a 30%-40% increase in generation time, while regenerating twice (three stories) resulted in a 50%-60% increase. The rate of performing regeneration decreased very slightly with the increase of ω.

With infilling, the change of ω did not change perplexity much (Figure 6.11b). This might be because infilling considers both beginning and end contexts, and would diverge less from the gold standard sentences. On the other hand, for continuation, as there is no end context to consider, the generated text can diverge significantly from the gold standard sentences. The steering difference and regeneration rate did not have a discernible trend with changes to ω. Without regeneration, the steering difference of infilling was larger than that of continuation. Similar to continuation generation, the addition of regeneration leads to decreased steering difference. The rate of regeneration was 40%-50% when regenerated once and 50%-65% when regenerated twice, which was higher than those of the continuation.

From the results, I find that a small ω value did not hurt steerability much. Moreover, a low ω value assured low perplexity in continuation tasks. If regenerating once, the steering difference decrease at a cost of 30%-50% more generation time. Considering the findings, in *TaleBrush*, I use ω value of 1 and try regeneration once as a baseline. Recall that I still limit regeneration to at most twice based on the writer's input. While additional regeneration may improve the results, the response time costs may be too high for an interactive application with the current architecture.

6.6.3 Human Evaluation on Generation

In addition to the automated test, I conducted a human evaluation to see if stories generated by *TaleBrush* are perceived as coherent, novel, and steered. For steerability, I showed evaluators two sentences generated with two different fortune levels and asked them to decide in which sentence the protagonist has a better fortune. Evaluators were not shown the input fortune values. If an evaluator chose the sentence generated with a higher fortune steering input value, I considered the answer correct. All fortune values in the human evaluation were randomly sampled. However, in the steerability study, I excluded steering input values that were too similar (a difference smaller than 0.25), as such similar steering inputs would be even difficult for humans to differentiate. I decided this threshold as 0.25 as fortune level differences larger than this value would be discernible to evaluators (i.e., five-level Likert scales are frequently used for sentiment annotation). I additionally investigated the novelty of generated texts as *TaleBrush* is designed to aid story ideation. I showed evaluators two versions of generated stories: one 'with' steering (i.e., using the GeDi model and character fortune values); and the other without steering (only with the prompt-tuned GPT-Neo model). I asked the evaluators to decide which story was more coherent or novel. For coherence, I specifically asked: 1) if story flow is coherent (story coherence) and 2) if sentences are grammatically coherent (grammaticality). While there can be more specific incoherent patterns [145], I chose story coherence and grammaticality as they each can tell us about the global

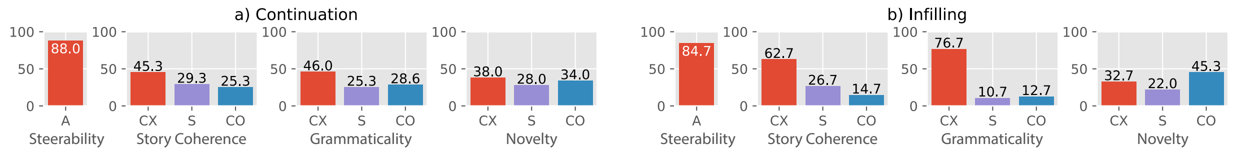


Figure 6.12: Human technical evaluation results on a) continuing and b) infilling generation. For steerability, the accuracy (A) in estimating which sentence is generated with a higher steering input value is reported. For coherence, grammaticality, and novelty, the ratio of evaluators choosing which condition has better quality is reported (CX for generation without steering, S for quality being similar, and CO for generation with steering).

and local coherence of the story. Evaluators could choose one of the generated stories with better coherence or novelty (CX for "without steering" and CO for "with steering" in Figure 6.12), or decide that they are similar in quality (S in Figure 6.12). Evaluators were not aware of which story is generated with fortune steering. For each criterion, I evaluated 100 story pairs, 50 for continuation and the other 50 for infilling. When generating these stories, I applied the regeneration strategy once and used $\omega = 1$. I hired crowd workers as evaluators, as I wanted to know if even non-experts in story writing perceive *TaleBrush*'s generations as coherent, steered, and novel. For each assessment question, I assigned three crowd workers to assure reliable evaluation. I recruited crowd workers from Amazon Mechanical Turk, who are in the US, with acceptance rates higher than 97% and more than 1000 tasks accepted. Each worker evaluated one criterion, for eleven pairs of stories including one gold standard question. I paid them \$3, which is over a \$10/hr payment rate.

Figure 6.12 summarizes our results. Evaluators recognized which sentence is generated with a higher fortune steering input value 88.0% of time for continuation (84.7% for infilling). Stories generated without fortune steering (CX) were chosen most frequently to have better story coherence and grammaticality. In continuation, their ratio was below 50% (45.3% and 46.0% for story coherence and grammaticality, respectively). That is, when using continuation with fortune steering, in more than half of cases, users would observe stories with similar (S) or better (CO) story coherence and grammaticality compared to those generated without steering. With infilling algorithms, similar to continuation, stories generated without steering (CX) showed better story coherence and grammaticality most frequently, but their frequencies were higher than continuation (62.7% and 76.7% for story coherence and grammaticality, respectively). One potential reason for this result might be the fundamental difficulty in generating coherent infilling stories when a given fortune sequence is not plausible (e.g., drastic fluctuation in fortune). For novelty, with continuation algorithm, different conditions showed similar performances. With the infilling algorithm, stories generated with steering (CO) are perceived to be more novel more frequently. In summary,

Table 6.1: Participants of the user study. Their domain of interest and years of experience in story writing is shown.

Expertise		Domain	Year	Expertise		Domain	Year
Novice	P1-5	N/A	N/A	Hobbyist	P10	Fantasy	2
Hobbyist	P6	Play script	2	Expert	P11	Sci-fi feature films	50
	P7	Time travel story	3		P12	Literary/Psychological fiction	15
	P8	Sports story	0.5		P13	Fiction, Fantasy	4
	P9	Hard SF, Dungeon & Dragon	3		P14	User/Personal/Fictional stories	8

I found that novelty is not hurt with fortune steering. I also found that our steering approach is reliable enough to steer the generation according to the steering input value, but at some cost to story coherence and grammaticality. The cost is lower in the continuation algorithm than in infilling one. Regardless of the strategy, no generative algorithm can generate perfect coherence or grammaticality. Thus, it is critical to ensure that the writer can edit the final text.

6.7 User Study

I conducted a user study to gain insights and feedback on the potential, limitations, and future opportunities of sketching interactions for iterative human-AI story co-creation. I focused on learning how our novel sketching input facilitates iterative exploration of story ideas and story co-creation. As this study seeks to understand how users leveraged our novel interactions, I did not conduct a formal comparison to existing human-AI story co-creation tools.

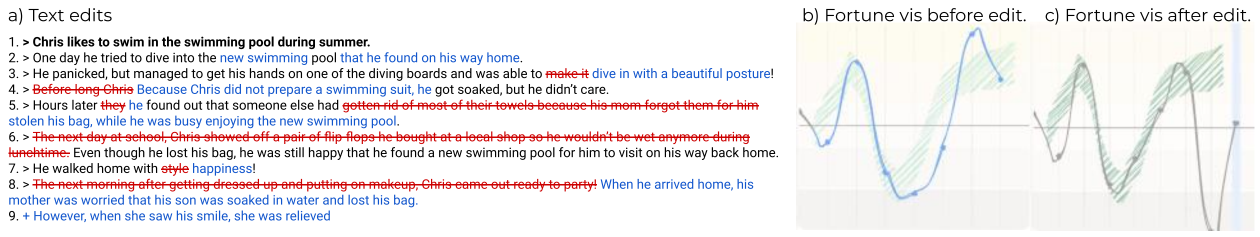
6.7.1 Participants

I recruited 14 participants (7 female and 7 male) by word-of-mouth and through online advertisements on social media and communities of universities. Participants completed a pre-survey on their background before the study. My participants had a range of expertise: novices, hobbyists, and experts. I classified participants who do not write stories occasionally as novices. Others were classified as either hobbyists or experts, and experts considered themselves as story-writing professionals. Participants were fluent in using English. I detailed participants in Table 6.1.

6.7.2 Procedure

I conducted a remote study with Zoom[2]. Participants were first given an overview of the study (15 min), then went through a tutorial on the tool (15 min). Participants could access *TaleBrush* directly through a URL we shared. In the tutorial, I explained that better fortune was represented

[2]https://zoom.us/



1. **> Chris likes to swim in the swimming pool during summer.**
2. > One day he tried to dive into the new swimming pool that he found on his way home.
3. > He panicked, but managed to get his hands on one of the diving boards and was able to ~~make it~~ dive in with a beautiful posture!
4. > ~~Before long Chris~~ Because Chris did not prepare a swimming suit, he got soaked, but he didn't care.
5. > Hours later ~~they~~ he found out that someone else had ~~gotten rid of most of their towels because his mom forgot them for him~~ stolen his bag, while he was busy enjoying the new swimming pool.
6. > ~~The next day at school, Chris showed off a pair of flip flops he bought at a local shop so he wouldn't be wet anymore during lunchtime.~~ Even though he lost his bag, he was still happy that he found a new swimming pool for him to visit on his way back home.
7. > He walked home with ~~style~~ happiness!
8. > ~~The next morning after getting dressed up and putting on makeup, Chris came out ready to party!~~ When he arrived home, his mother was worried that his son was soaked in water and lost his bag.
9. + However, when she saw his smile, she was relieved

Figure 6.13: P2's work: a) Edit of a generated story. Bolded, red, and blue text indicates initial context input, removed parts, and added texts, respectively. b) Line sketch and fortune arc of the generated story before editing. c) Fortune arc of the edited story.

with higher points on the y-axis. However, I allowed participants to use the tool to specifically learn how the mapping worked as there are no explicit axis labels. After the tutorial, I asked participants to use *TaleBrush* in two tasks. First, they were given a specific story beginning and were asked to generate a continuing story with *TaleBrush* until they found one that appealed as a draft story. After participants picked this draft story, they were asked to edit the text directly without additional generation (20 min). This task allowed us to observe how participants drew fortune arcs and which of the generated elements they retained or modified. For the second task, I asked participants to freely use the tool, bringing in their own story from the very beginning (20 min). I did not restrict how and when they used generation. Whereas the first task roughly corresponded to one 'turn' (machine then human), the second allowed us to observe an iterative co-creation process. In both tasks, I asked participants to share the screen of the interface. I also asked participants to think aloud while using *TaleBrush* to learn their rationales and reactions in using the tool. Lastly, I conducted a short interview, asking the experience of using *TaleBrush*, such as how they used the line sketching, how they incorporated stories generated by the tool, and how they would adopt *TaleBrush* to their practice (20 min). The whole session was video-recorded.

6.7.3 Results

I qualitatively analyzed screen recordings, think-aloud statements, and interviews. I analyzed data by iterative coding with inductive analysis, and coded results were reviewed with two other authors. Note that unless explicitly stated, results are about the participants across all expertise levels.

6.7.3.1 Use of Line Sketches

Line sketching facilitates iterative co-creation with intuitive steering. Overall, participants indicated that line sketching was a simple and expressive way to specify the protagonist's fortune. As critically, they understood how to use sketching to steer text generation. Participants also thought that fortune was an effective attribute to steer the overall story flow. During the study, writers experimented and iterated on multiple line sketches. Some tried generation on a sub-part of the story to only change that part. Others tried generation in the middle of editing the story. For example, P6 repetitively tried generation to find infilling sentences that would go along well with the ending sentence previously generated from *TaleBrush*. During the process, P6 also edited generated sentences and used them as the context for the next generation. Participants also mentioned that sketching allowed them to specify the arc without the pressure of being precise. For example, P6 mentioned: *"If I should have input each number, I might have been more reluctant to complete the story. Because they are specifying more detailed values. So, I think drawing would be more helpful in generating the stories."*

Surprise control and steerability in small changes could be improved. A few participants used the surprise control, but the perceived effectiveness diverged between participants. Additionally, when participants used sketching for small and detailed fortune changes, they felt that their control was not reflected well on the generation. The feature of specifying steering fidelity with sketching speed might have been one approach to solve this problem, but it was not effective enough in some cases. Participants speculated that the feeling of lack of steering might be due to disagreement between their perception of fortune and the fortune level calculated by *TaleBrush*. The current generation approach of replacing the whole sentence may have also exacerbated this problem, as the user expected a slight change in the text when they are making small changes in the sketch. While the new sentence may be at the appropriate fortune level, the content may have been too radically different from the sentence at that time point.

Line sketching facilitated the planning of writing to some participants. Within the iterative co-creation process, the line sketch could also influence participants to edit the story to follow the sketched fortune arc. For example, in Figure 6.13, P2 edited generated text to follow the line sketch that P2 drew in most sentences (from b to c). Because the current story was constantly compared to the drawn sketch, the participant would try to adjust the story so that it was more aligned with the sketch. Participants also mentioned that the sketch served as a plan or a guideline. For example, P7 mentioned: *"As I draw graphs, it is intuitive, I could easily see how I would compose the storyline and how I should structure it."* However, some participants who deeply focused on the story itself did not consider the sketch when editing the text.

6.7.3.2 Use of Generated Texts

Generated texts are used as ideation materials. From the automatically generated stories, participants adopted characters, settings, short expressions, sentences, events, or even the overall story flow. For example, in Figure 6.13a, P2 liked the event described in the third sentence and edited the story to go along with the sentence. Novice participants mentioned that generated texts lowered the barrier on starting writing, as they would have more frustration if they start from a blank paper.

Incoherent generations are targets for revision. Unsurprisingly, participants frequently revised incoherent parts. For example, the sixth sentence of Figure 6.13a was taken off because mentioning flip-flops was perceived as a drastic context change. Participants also added more details to the story to make it more reasonable and concrete. For example, in Figure 6.13a, P2 added details on the 'pool' in the second sentence that it is a new swimming pool on the way to the protagonist's home. Participants also added new sentences to make the story flow natural. For instance, in Figure 6.13a, P2 added the last sentence to make the ending more natural. Interestingly, the editing pattern of novices, hobbyists, and experts did not differ much, potentially due to the novelty of the tool's functionality to participants.

Incoherent generations serve as the reason to exert creativity for some participants. While incoherent or ambiguous story elements require participants to revise sentences, some participants appreciated this as a 'prompt' to apply their own creativity and revise the text. For example, P1 mentioned: *"If, there is nothing like that [incoherence], I would not do something creative. However, with something that needs to be fixed, and if you kind of get that they need to be revised, I feel like I can exert my creativity... So I don't think these are downsides."* Moreover, some 'incoherence' was perceived as novel elements to participants. This 'bug' thus becomes a 'feature' in the context of ideation support.

6.7.3.3 Potential of *TaleBrush* in practice and suggestions for improvements.

Potential for quick and iterative ideation. Experts and hobbyists mentioned that they would use *TaleBrush* to quickly iterate through diverse ideas. Collected ideas can be used in the ideation stage or to overcome writer's block. This benefit would be maximized in settings where the writer should bring up multiple stories, such as tabletop role-playing games.

Supporting more attributes, longer texts, and various types of writings. Participants mentioned that they would like to see steerability for more story attributes, such as other characters' fortunes, settings, or genres. They reasoned that more steering would likely lead to a more desired text that they would adopt to theirs. Some participants indicated that the visualization and steering granularity might need to change if *TaleBrush* expands to generate longer texts, as users would want to give paragraph-level or chapter-level specifications. Experts noted that *TaleBrush* needs to be adapted to each usage context. For example, P11, a screenwriter, stated that the tool should be able to express character emotions or states only with observational statements, as those need to be shown, not told, in screen writings. Finally, participants mentioned the potential of using sketching to steer generation of texts other than stories, such as steering tone in speech or editorials.

Addressing bias. Potential bias in the story generation and steering was brought up as an issue. For example, P5 perceived that the algorithm generates stories that are more likely written in western culture. P10 mentioned that the steered generation did not align with what they expected when the protagonist is an anti-hero, signaling potential bias. An important starting point would be in ensuring a more diverse training set, as is broader testing.

6.8 Discussion

From *TaleBrush*, I reflect on lessons learned for steerability design and iterative human-AI co-creation.

6.8.1 Sketching to Address The Gulf of Iterative Human-AI Co-Creation

The use of sketching as interactive steering inputs addresses challenges in the gulf of execution and evaluation [299] between the user and the generative AI systems. A key feature is reducing the friction in the iterative co-creation process. First, the **steering interaction** of *TaleBrush* is designed to be *expressive* and *easy*. These are often conflicting goals: simple interfaces have limited expressiveness and expressive interfaces are not simple. However, this balance is clearly needed for closing the gulf of execution but is missing in many existing interactive approaches to co-creation. Second, to minimize the gulf of evaluation, **sensemaking** of generated results should be facilitated. Users should be able to more easily understand how the given steering inputs relate to what the AI generated. This is often missed in 'explainable' AI approaches. The explanation does not map to either the user's intent or how they interface with the software.

6.8.2 Generalizing Line Sketching-based Steering

Beyond story generation, line sketching can potentially be adopted for other applications that generate content with sequence attributes. This is particularly true in domains where we can operationalize and encode features of the content as drawings. For example, poetry [36] and lyrics [415] can be visualized based on moods or rhythms. One indicator of whether the line sketching approach is appropriate for a generative context is if time-series style visualizations already exist in that medium. For example, in audio, we can visualize (and partially steer) changes in volume over time. This can be extended to support more powerful types of guidance. For example, a composer could steer rhythmic density or melodic contour [4] of generated outputs. Similarly, a sketch line could be used with a text-to-speech generator for flexible micro expression control (e.g., pace or aggressiveness) [352].

6.8.3 Disagreement Between Human and AI

In experiments, I found situations in which the steered generation do not satisfy a user's expectation, and the generated text's fortune level might not accord with the given steering inputs. This can be due to algorithmic error, either from the steered generation or recognition. However, I found that in some cases this was a result of an implausible request (e.g., drastic fluctuation of fortune). Dramatic fluctuations are not often observed in real text and the generative algorithms may not be able to generate suitable text. To address this, we might limit how much the sketch can move between time steps (e.g., a smoothing). Alternatively, we can leverage the ambiguity and undetermined nature of sketching [144, 229] to communicate uncertainty in steering. In *TaleBrush*, as the user sketches, it shows a wide range where the generated output would likely fall. While these design elements manage user expectations on the tightness of steering, the user study found that users would want more sensitive steering when they iterate over generations with small changes in parameters. The sketch-speed approach is one way to allow users to encode how 'tightly' they want the story to fit. However, it may be worth considering alternative approaches in the future (e.g., explicitly selecting different 'pen' widths or experimenting with things like pen pressure).

As the user study revealed, the writer's perception of the protagonist's fortune can also be misaligned with how the tool recognizes it. Adapting the machine's fortune scale to the user's perception can be a way to solve this. For example, we can allow users to revise the machine recognition results based on their perception of the text (e.g., by moving the line the machine drew). This can be fed back to the underlying classifier to learn the user's scale of fortune. To enable such interactions with low user effort, it should be technically possible to adjust models with few data instances from the user. Recent language model tuning approaches, including soft prompt tuning [235, 251], may support this approach. In some situations, the user's expectation may be mismatched due to the potential bias in the model or training data. For example, the study participants mentioned the model seemed to show biased behaviors, such as generating stories more likely written in western cultures, which might be due to skewed distribution in the variety of stories [27]. To alleviate this problem, we can try pretraining big language models with balanced data [27] or having a steerable module that tries to control these biases [207, 246].

6.8.4 Further Support in Story Co-creation

Story attributes other than the protagonist's fortune (e.g., conflict, settings) can also steer the story flow. To decide which attribute to support, the writer's specific needs should be understood first.

Based on the identified attribute, visual steering inputs can be designed to assure expressive and simple steering for the attribute. For example, if writers want textual keywords to be used as a steering input [430], placing keywords along the story sequence axis can be one design option. Alternatively, if a writer wants to convey condensed information about character interaction, diagramming a character network graph can be a steering option [14]. For example, one could imagine drawing multiple instances of a network diagram to guide the AI by which characters in a story should interact and how. Alternative sketched representations are worth studying both for text generation [282] and beyond. To support these, the dataset and algorithmic pipeline would need to be expanded based on the attribute to be steered.

For specific story domains, such as screenwriting, generation can be extended to adopt the "show, not tell" approach. For such extensions, the dataset would need to be constructed with screen scripts while the algorithms would also need to consider the latent fortune states throughout the story.

TaleBrush can potentially be extended to support the authoring of high-fidelity, longer text. To provide such support, the tool would need to consider writers in the translation stage [111, 142], where ideas and plans are transformed into detailed text. Hence, generative tools should be able to follow the writer's specifications. The annotation approach would also need to be reconsidered, such as annotating the fortune of the character after summarizing the long text with crowdsourcing [406] or algorithms [26]. Moreover, the steering and sensemaking should be re-designed for longer text. For example, a tool might allow the writer to first sketch an outline and then a detailed story [104]. A more complete tool might enable a long-term study of how co-creation impacts writing.

6.8.5 Limitation and Future Work

In this chapter, I qualitatively investigated how writers would use *TaleBrush* and sketching interactions. Future work can investigate how proposed interactions impact the story writing compared to writers' current practices or when steered generation is used without sketching interactions. Moreover, I did not look into how each function impacts story writing separately. For example, how would the control of surprise impact the writing experience alone? To answer such questions, more controlled user experiments would be needed.

While my annotation approach produced usable data, I identified ways it could be improved in the future. While collecting comparison information with the slider interface, I saw that how calibrating sentences were annotated would impact the annotation results. For example, if one calibrating sentence has been annotated with maximum fortune and the annotator later realizes that one of the task sentences has higher fortune, the annotator would either not annotate accurate comparison information or need to re-annotate the calibrating sentence. One possible approach is to modify the interface so that sliders can be extended without having maximum or minimum ends.

Finally, while the training data specified which character's fortune should be steered, the trained model frequently coupled multiple characters' fortunes. For example, in the story about Bob and Alice, it is hard to make Bob have a good fortune and Alice have a bad one. Explicit steering of multiple characters' fortune can be one future work direction to handle this limitation.

6.9 Conclusion

I introduce *TaleBrush*, a human-AI story co-creation tool that allows writers to sketch out textual stories by visually steering the protagonist's fortune. The visual sketch is used as an input to the GPT-based story generation model and the model generates stories according to the specified protagonist's fortune. To help make sense of the generated story, the protagonist's fortune is visualized using the same abstract representation as the drawn sketch. From a technical evaluation and a user study, I found *TaleBrush* has reliable steerability, while its sketching interaction facilitates participants to iterate on generation to collect novel ideas that align with their intentions. With the recent advances in generative ML algorithms, I hope *TaleBrush* opens up new ways to provide frictionless steering. Connecting back to the design approach in Chapter 4, this chapter shows how we can facilitate effective and efficient expression with iteration in steering interactions by adopting an alternative, but familiar input modality of visual sketching.

CHAPTER 7

PromptPaint: Steering of Text-to-Image Generation Through Paint Medium-like Interactions

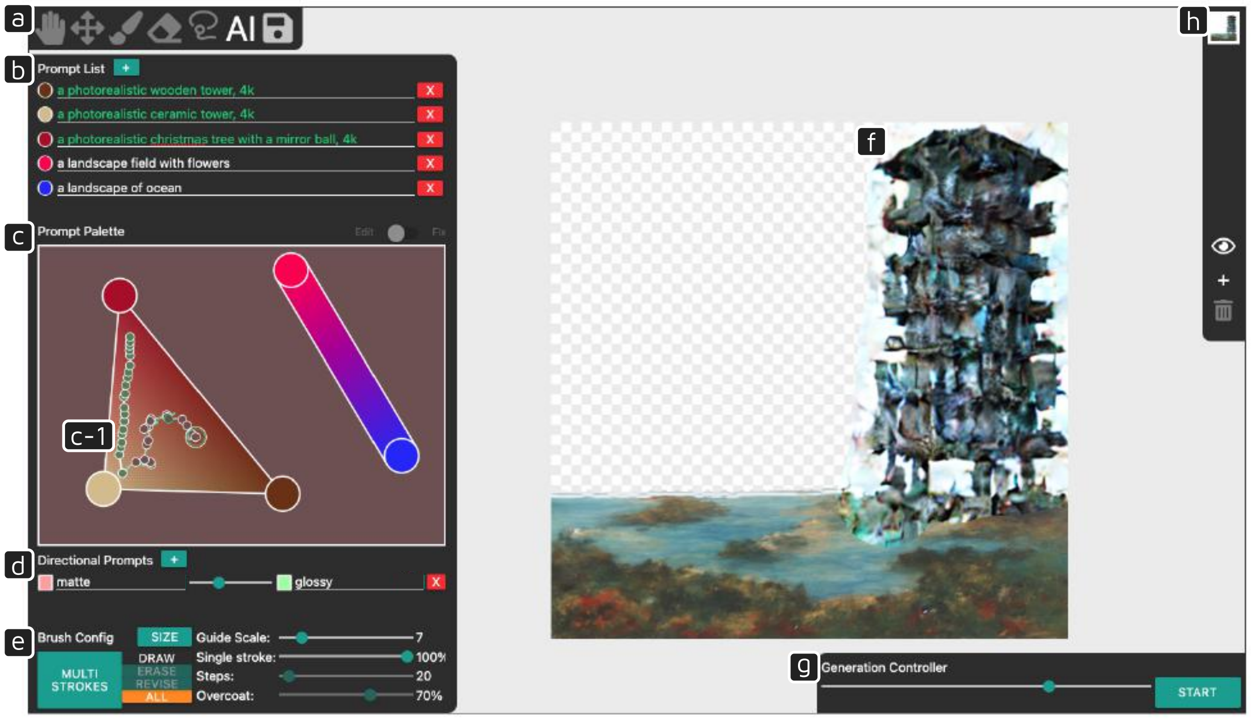
7.1 Introduction

Text-to-image (T2I) models that generate high-quality images from textual prompts are rapidly advancing with diffusion-based techniques [298, 330, 343, 348], inducing a lot of excitement about democratizing the creation of visual images. Input interactions of textual prompts allow users of algorithms to specify their intentions regarding which image to generate while minimizing the required level of skills (e.g., drawing). These simplistic interactions and high-quality renditions, along with accessible tools (e.g., Dream Studio[1], Gradio demos[2], demos in Google Colab Notebooks[3]), allowed a large set of the public without visual art expertise to generate visual images.

Despite the excitement, to use these models to support human creation, there are limitations in how they function. One problem is that people often create visual images with a procedure, while these models do generation in an end-to-end fashion. For example, people often follow specific workflows to create visual images [88, 98, 433]. For example, in digital comics, artists draw the piece in order of sketching, flatting, shadowing, and adding backgrounds and special effects [433]. Creation procedures can involve intermediate decisions between steps. Unlike existing practices, images generated only with the user's initial prompts would limit what the user can do during the artifact generation. Another problem is that natural language prompts are not enough to express all of what users would want—sometimes their targets can be ambiguous (e.g., targeting a style with elements of both Impressionism and Arte Nouveau). Natural language prompts would also be inadequate when the user needs to iterate on the generated images with a certain degree of changes (e.g., "a bit less angular shape").

With these limitations, researchers and practitioners recently introduced technical approaches

[1]https://beta.dreamstudio.ai/
[2]https://gradio.app/
[3]https://colab.research.google.com/



Guide Scale: 7
Single stroke: 100%
Steps: 20
Overcoat: 70%

Figure 7.1: *PromptPaint* allows flexible steering of diffusion-based text-to-image generation by combining prompt-based generation with paint medium-like interactions (e.g., oil painting, watercolor). After the user defines their prompts of interest (b), the tool allows users to mix prompts like how we mix colors of paint mediums by 1) interpolating them on the prompt palette (c, *prompt mixing*) or 2) adding extrapolating attributes (d, *directional prompt*). Moreover, *PromptPaint* allows gradual shaping of the artifact by allowing 1) changing of prompts during generation (c-1, g, *prompt switching*) and 2) spatial selection of area to generate image content (f, *prompt stencil*). *PromptPaint* also allows detailed specification of generation with configuration widgets (e) along with other image editing functions like moving image content, brushing, erasing, lassoing, and layer edits (a, h).

to enable gradual editing of visual content. An approach is to in-paint and out-paint, adding or revising visual elements on the existing image [20, 298, 347]. Another set of research investigated image-to-image approaches, which allow users to give an initial image to build up on the generation [20]. The researchers also investigated technical approaches to mix prompts [240, 249] or edit images based on natural language prompts [72, 159, 203, 404]. While technical approaches and corresponding demos exist, there have been few considerations on how the interactions should be modeled to facilitate gradual creation experiences with more controls in the user's hand.

In this work, I explore approaches for users to interact with T2I models that enable the gradual building of artifacts while allowing flexible exploration in possible art space. I suggest the idea of considering interactions with T2I models as how we interact with paint mediums (e.g., physical ones such as oil paint or watercolor). First, we are familiar with the interactions with paint mediums, as even novices in visual arts have some experience learning to play with them [98]. Second, users can manipulate paint mediums to enable various expressive renditions. For instance, we can mix them to get colors beyond what we have in color tubes and apply different colors to different parts of a canvas to create a unique image. By bringing such interactions into T2I models, I expect to enable more gradual and explorative creation even with AI generations. With these high-level ideas, I implement *PromptPaint*, a tool that turns prompt interactions into familiar interactions with colors of paint mediums. With such interactions, *PromptPaint* 1) turns prompts into flexible materials that can even target verbally indescribable concepts with *prompt mixing*, similar to how we mix paint medium colors, and 2) modularizes image generation by allowing users to apply varying prompts to different parts of the canvas (*prompt stencil*) and different parts of the generation process (*prompt switching*), similar to how we apply different colors onto the canvas.

From the user study, I found that different ways to steer T2I generation could allow users to generate images that align well with their intentions through iterations. However, I also identified design trade-offs of 1) focused iteration and curation and 2) manual editing and automation. Furthermore, the high complexity and randomness of AI models could result in a misalignment between AI behaviors and user expectations. Lastly, while users had some sense of ownership of the resulting artifacts with their contribution to high-level ideas, I found that the user's expertise and the result being aligned with the user's expectations can impact their sense of ownership. From the findings, I discuss insights on adopting paint-medium interactions in designing future versions of generative tools.

7.2 Background and Related Work

7.2.1 Painting-From Physical to Digital

As *PromptPaint* is inspired by the interactions with paint mediums, I first describe the characteristics of the painting that *PromptPaint* borrows from. Painting is one of the means of rendering visual images by applying paint medium onto a canvas [98]. This "personal causation" [81], or the change in the world by an individual, is an intrinsic satisfaction that can come from doing a painting. Painting enables us to express ideas and emotions not describable in other mediums, such as poets or music [98]. Moreover, painting often involves continuous judgments during the process[98]. When people can routinize the creation of a specific set of artifacts, people would structure such judgments as a workflow [88, 433]. These characteristics hint at the limitations of existing T2I models, while showing the potential benefits of leveraging painting interactions. First, not all visual ideas would be describable with natural language prompts, and paint-medium-like interactions can potentially alleviate such limitations in natural language prompts. Second, as painting interactions often involve gradual judgments (and potentially iterations), those can guide us on how to design gradual and iterative interactions with T2I models. Third, paintings-like interactions would likely lead to “personal causation” and intrinsic satisfaction. The idea of applying painting interactions in T2I models also resonates with enabling direct manipulation in intelligent interfaces, by balancing it with automation [366]. With such benefits, I ground the design of *PromptPaint* on how we interact with paint mediums.

There has been a line of work for tools to support painting and drawing. Although often implemented for digital painting, some tools guided novice users with visual guidance, without directly intervening in user drawings [183, 421]. Other types of tools tried to augment the novice user’s skills by adding corrections to the drawn result [108, 241, 387, 429]. There have also been tools such as Color Builder, which allows users to flexibly explore color options with color mixing interactions [368]. Although the tools mentioned above are designed more to support “existing drawing/painting practices,” there have also been tools that enabled users to generate novel types of artifacts with computationally enhanced brushes [28, 186, 356]. *PromptPaint* builds upon existing tools by bringing diffusion-based T2I models closer to interactions that people would have with paint mediums.

7.2.2 AI Image Generation

Although I focus on diffusion-based T2I models, there has been a history of approaches that generate images with neural networks. One of the first approaches that caught researchers’ attention was style transfer algorithms, which can transfer the visual style of an image to another image [129]. This technique was enabled by decoupling content and style representations from convolutional neural networks (CNN) [226] and combining those representations from different images. These algorithms advanced to allow multiple [91] to arbitrary [360] sets of images as a style input but had limitations in that they focused on transferring styles and could not generate images from scratch. Another set of approaches that can generate images is generative adversarial networks (GAN) [139]. GANs train a generator model to fool a discriminator model that classifies whether or not the generated result is fake. GANs also use CNNs for the generator and the discriminator and could generate images from scratch. Researchers further research these models to condition the generation to specific classes [62]. The most recent approach is diffusion models, which learn to recover images from noisy images [162]. These models were capable of generating high-quality images compared to other approaches. With these models, researchers also devised approaches to guide their generation, with classes [86, 163].

Parallel to the aforementioned approaches, researchers also trained representations that combine text and images. One of the most noticed representations was CLIP [327], and this shared representation enabled natural language-based guidance for image generation approaches [73, 227]. Among them, diffusion-based T2I models have attracted a lot of attention with their flexibility to follow input prompts and high quality [298, 330, 348]. However, these have limitations in that generation occurs end-to-end. Hence, imbuing more human intentions into the generated results can be challenging. To address such a problem, researchers investigated various approaches, from seeding an initial image that would be transformed [20] to combining two different prompts to realize them in the image at the same time [249], generating an image of one prompt while having the overall form of another prompt [240], editing or expanding images with visual masking [20, 298, 347], and allowing editing of images with natural language prompts [72, 159, 203, 404]. Although these introduced technical approaches to gradually and iteratively shape images, there is not yet a good model of how users could interact with diffusion-based T2I models. In this work, I investigate interaction approaches for T2I models with an analogy to how we use paint mediums.

7.2.3 Interaction with AI Generation

There have been numerous approaches to interacting with AI generations in various domains. An aspect to consider was how to *steer* the behaviors of these generative models. Less flexible options would be to use rigid categories as input [62, 223]. The limitation of this approach is that since there are a limited number of options that the user can choose, the space to explore is limited. An alternative option would be to receive specific examples as input, which are often used for algorithms such as image style transfer algorithms [307, 360]. Although technically flexible in receiving "any examples," interaction-wise, iterating on steering can be challenging, as searching for another example with a slight difference can be difficult. To support more flexible steering iteration, slider-based controls have been devised for domains including image [74, 310], and music [256] generation. Such control interfaces allowed users to flexibly iterate their input within a defined spectrum of space. Researchers extended the continuous space of controls to interfaces with more flexible control, such as explorable gallaries [443] or visual sketches [66]. With advances in language models [44] and contrastive learning between text and other mediums [327, 425], natural language prompts also became another means to steer AI generation. For example, with the ability of language models to support a wide range of tasks, researchers introduced tools to help users manage prompts [386, 427]. Prompt-based generation extended to other mediums than texts, such as images [330, 348], UI design [214], codes [58], 3D models [188, 322] and even videos [373, 407]. The prompt-based generation has a comparative benefit over other approaches in that it does not limit input made by the user (apart from whether the model can reliably handle those). However, at the same time, prompting can be challenging, since 1) there could be too many options to explore, and 2) some concepts are difficult to verbalize due to ambiguity and vagueness (e.g., “a bit less vivid color”). In this work, inspired by the interactions we do with paint mediums, I extend existing interaction approaches for generative models to the mixture of prompting and visual interactions.

Another aspect to consider in interaction design for generative models is how the generation processes and results are embedded into the human art creation process. For language generation, generation occurs in token-to-token fashion [405]. The generated results can be easily repurposed as a partial material that can be used during the user creation process. However, for other types of medium (e.g., images, audio, or videos), advanced models were often designed to generate artifacts that tend to be the final version. In such cases, designing a tool that allows users to adopt generated results during the human creation process would be more challenging. In diffusion-based T2I model contexts, researchers and practitioners investigated ways to repurpose generation results as a *modularized* unit in the human creation process. As introduced in Section 7.2.2 in-painting, out-painting, image-to-image, and prompt-based editing generations would be specific examples. In this work, in addition to repurposing the generated *results* as a modularized unit in human creation, I also investigate the modularization of generation *process* by allowing user interaction during the process.

7.3 Interacting with Generative Models Like Paint Medium

In this work, I extend the ways of interacting with generative models by using the analogy of interactions with paint mediums (Figures 7.2 and 7.4). The extensions focus on two different aspects. The first is allowing users to go beyond discrete semantics (e.g., categories, prompts) for specifying generation and explore spaces in-between semantics in the vector space. With the paint-medium analogy, we connect this manipulation to color-mixing interactions. The second aspect is allowing users to gradually generate artifacts, similar to how we gradually apply colors when we are painting. In the following sections, for each aspect, we explain detailed connections between the steering of generative models and paint-medium interactions.

7.3.1 Mixing Colors: Exploring Vector Spaces

Input modalities of categories, examples, and prompts are intuitive interfaces to specify the semantics of artifacts to be generated, as they are in a comprehensible format for human users. When used in generative models such as language models [66, 223], style transfer algorithms [360], GANs [6, 198], or diffusion models [330, 348], they are first transformed into a vector representation to guide the generation of those models. This indicates that these discrete semantics can have limitations, as vector representations in-between discrete semantics would be hard to reach only

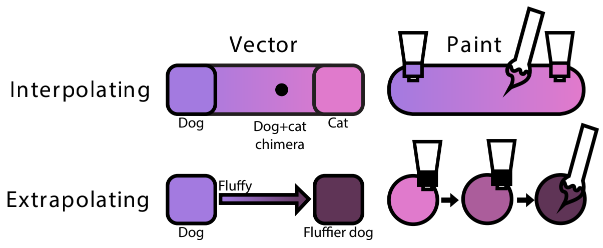


Figure 7.2: Mapping interactions of exploring vector spaces to interactions of mixing paint colors. Each of the discrete semantics (which would be categories, prompts, or examples and are represented as black squares in Vector) can map to discrete colors in paints (tubes in Paint). Then, the user can explore vector semantics between discrete semantics (black dots in Vector) in a way similar to how they would explore colors by mixing (brushes in Paint).

with these interfaces. However, there may be cases where users would want to further explore such semantics. For example, when using prompts, if the user wants to explore concepts that are difficult to be verbally described, they might want to explore those in-between vector representations. Such needs would also arise when the user wants to do fine-grained control in certain aspects, like adjusting the roughness of texture in image rendition. Technically, previous work has shown that such manipulation is doable with interpolation of discrete semantics [66, 328] or extrapolation by identifying directional vectors for manipulating concepts [310, 353]. However, not many previous efforts have tried to turn such manipulation into accessible interactions, specifically for the case when the user is capable of flexibly using different discrete semantics, like prompts.

To facilitate the exploration of in-between vector representations, we introduce the idea of interacting with discrete semantics in a similar way to how we *mix* paint medium colors. When we are using colors to paint a canvas, those different colors come into our hands in the form of tubes, each having one discrete color. However, when applying them to the canvas, we do not limit ourselves to those discrete colors. We go beyond the given colors by putting them on the palette and mixing them. Similarly to the color mixing interaction that occurs in the color palette, we introduce the interaction of mixing different discrete semantics on *semantic palette*. Analogically, each discrete semantic of categories, examples, or prompts would map to a discrete color for the paint mediums (Figure 7.2). The first specific approach to mixing discrete semantics is to *interpolating* them, by mixing two or more different discrete semantics on the semantic palette and exploring the space between them (top row of Figure 7.2). For example, to render an image that is the chimera of a cat and a dog with T2I models, the user would be able to interpolate the semantics of a cat and a dog. It would be similar to spreading colors to mix them and seeing the gradient of colors in between when we are mixing colors. The interaction of interpolating semantics, hence, can be visually represented in interactions similar to those in the color palette. The second approach is to *extrapolating* them, adding a directional semantic to the semantic that the user is considering to

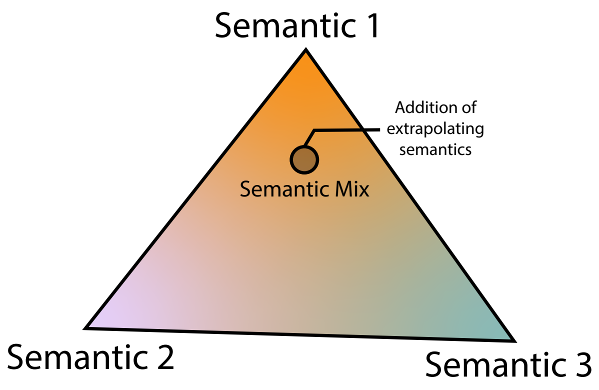


Figure 7.3: Using palette interaction for semantic mix has benefit in that the interpolation and the extrapolation can be represented in the same interface.

do a fine-grained control of it (bottom row of Figure 7.2). For example, with T2I models, the user can render an image of a dog with a certain level of fluffiness by adding the semantics of fluffiness to the semantics of a dog. In the use of paint mediums, it would match the interaction of adding a small amount of a different color to steer the characteristics of the color that we are dealing with (e.g., making the color darker by adding a bit of black pigment). Hence, the extrapolation of semantics also can be visually represented by adding a color pigment. Note that these interactions would be adoptable for those generative techniques that can turn user-interfaceable discrete input modalities into the vector space and that can perform generation reliably on the arbitrary vector space of the representations.

Note that there can be other interactions to mix discrete semantics than color palette interactions. For example, we can mix the prompts with the slider widgets. Compared to such interactions, palette interactions can represent the mix of discrete semantics with two visual signals, their positions, and colors, on the palette. With color representation, specifically, both interpolation and extrapolation can be shown in a single interface. That is, as in Figure 7.3, the palette interface can represent the interpolation with the selection of a point in the mixed-color gradient and show the extrapolation by adding the color to the selected interpolated point. On the other hand, with slider-like interactions, only weights for interpolation and extrapolation would be represented, not the final mix of semantics.

7.3.2 Applying Colors onto Canvas: Gradual Generation

Generation algorithms often generate artifacts without allowing user interventions. It can render the experience of using generative models far from "creation," where the user shapes and steers the artifact with their intentions. Here, I explain the potential interactions that can allow users to have more interventions in the generation process. Again, I take the analogy from how we interact with

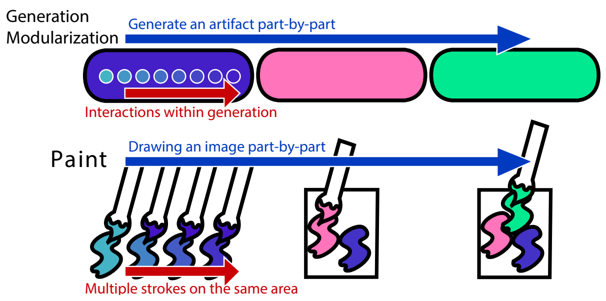

Generation Modularization
Generate an artifact part-by-part
Interactions within generation
Paint
Drawing an image part-by-part
Multiple strokes on the same area

Figure 7.4: Mapping interactions of generation modularization to interactions of gradually painting an artifact. Within-generation interventions would correspond to multiple strokes applied on the same canvas area, and generating the artifact part-by-part would map to drawing the image part-by-part.

paint mediums, focusing on how we apply them to the canvas. When we apply colors to the canvas, we do not simply use the same paint for the whole area. Instead, we gradually build the artifact with multiple paint strokes, sometimes overlapping them. While such a painting process would require far more effort and expertise from the user compared to using generation models, we can similarly modularize our creation into some units when using generative models (e.g., each stroke for visual arts or words/sentences for writing). This *modularization* allows us to repetitively intervene in the creation process, facilitating the steering of the generation. With generative models, there could be two ways of modularization. The first approach is to allow interactions within the generation process (red arrows in Figure 7.4), such as changing the guiding prompt during generation. In paint mediums, it would be similar to applying different paint strokes in the same area to decide the final rendition. This specific approach would only apply to those techniques that follow modularized procedures so that the user can intervene. The diffusion-based T2I model can be an example. As I will demonstrate in the later sections, in diffusion-based T2I models, those prompts that are used in the earlier stage decide the overall form of the image rendition while those in the later part decide the details. For example, using "a banana on the ground" first and then switching it to "a futuristic car" would result in a futuristic car in the shape of a banana. The second approach is to enable partial generation instead of generating all content with one generation so that the user can gradually build the artifact (blue arrows in Figure 7.4). Analogically, it would match how people draw an image part by part. Note that this interaction can be easier to implement for some generative models that generate part by part. For example, causal large language models generate texts token-by-token, which can be a part of the final artifact. For other models not necessarily designed to generate in part, there are also cases where researchers and engineers figured out that the model is capable of partial generation. In-painting and out-painting of diffusion-based T2I model can be examples [20, 298, 347]. For example, with T2I models, the user can first generate the overall background (e.g., the ocean scene) and specific objects later (e.g., boats).

7.4 *PromptPaint*: Interface

Based on the design approach introduced in Section 7.3, I design and implement *PromptPaint*, an image creation tool powered by a diffusion-based T2I model (Figure 7.1). This tool allows users to flexibly shape image generation with 1) exploration and fine-grained control of prompt space with *prompt mixing* and *directional prompt*, and 2) gradual building of visual images with *prompt switching* and *prompt stencil*. In this section, I introduce the interactions in *PromptPaint*.

7.4.1 Canvas and Basic Editing Functions

PromptPaint has a canvas where the user can create digital raster images (Figure 7.1h). It accompanies raster image editing functions that can also be found in other raster image editing applications, such as moving the position of the canvas, moving/rotating images inside the canvas, brushing, erasing, and lassoing (from left to right of Figure 7.1a, except the right two). Furthermore, the user of the tool can add more than one layer to the canvas, change its ordering, hide them, or even delete some (Figure 7.1h).

7.4.2 T2I Generation Functions

With diffusion-based T2I functions, the user can generate images on the canvas. The user first specifies and selects the prompts to guide the generation (*prompt mixing* and *directional prompt*). The user can then start to run the generation by specifying the area to put the generation results (*prompt stencil*). When running the generation, the user can also change the guiding prompts during the generation (*prompt switching*). We first explain how the user can specify each prompt and then describe each function. We give technical details in the later section.

7.4.2.1 Prompt List

To generate images, the user can add a list of prompts in the Prompt List widget (Figure 7.1b). The user can add a new prompt with the + button. Each added prompt has its own color and text prompt, which can be edited by the user. To change the color, the user can hold down the colored circle for a while to reveal the color picker. The user can edit the textual prompt with the text box. The user can also delete the prompt with the X button.

7.4.2.2 Prompt Mixing

Each prompt is rendered as a color circle in the Prompt Palette widget (Figure 7.1c). The user can position these prompts as they wish with dragging interactions. The user can also mix these

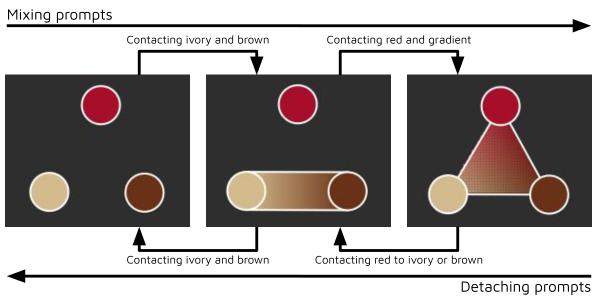

Mixing prompts
Contacting ivory and brown
Contacting red and gradient
Contacting ivory and brown
Contacting red to ivory or brown
Detaching prompts

Figure 7.5: Interactions to mix and detach prompts in Prompt Palette.

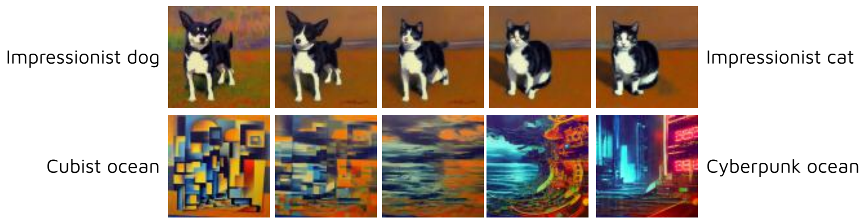


Figure 7.6: Example results of prompt mixing.

prompts, allowing users to flexibly explore the vector space between the specified "discrete" prompts. The user can perform *prompt mixing* by directly manipulating the prompt color circles, with an interaction similar to how we mix colors with paint mediums. For example, as in Figure 7.5, the user can touch one of the prompts on another to mix two prompts. Moreover, if the user wants to add another third prompt to the mix, they can touch the mixed gradient with the third prompt. While the current version of *PromptPaint* allows a maximum of three prompts to be mixed, future versions could be designed to allow the mixing of more or infinite numbers of prompts. If the user wants to detach a prompt from the mix, they can move the prompt to detach to touch any of the other prompts within the mix. For a generation input, the user can select one of the prompts or a point from a gradient of prompt colors. The selected point will be rendered as a circle (Figure 7.1c-2). For selected points, the prompts that are included in the selected mix will be highlighted in green on the Prompt List interface (Figure 7.1). The selection within the gradient would mix different prompts through interpolation and can be used to guide the diffusion model's generation. Figure 7.6 shows examples of mixing two different prompts with prompt mixing.

7.4.2.3 Directional Prompt

In some cases, the user might want to add more attributes to the generated results (e.g., adding more complexity in the interior design), similar to how we change color by adding other colors (e.g., making red darker by adding more black). *PromptPaint* supports such an interaction with directional prompts, which extrapolates the prompts to introduce additional attributes (Figure 7.1d).

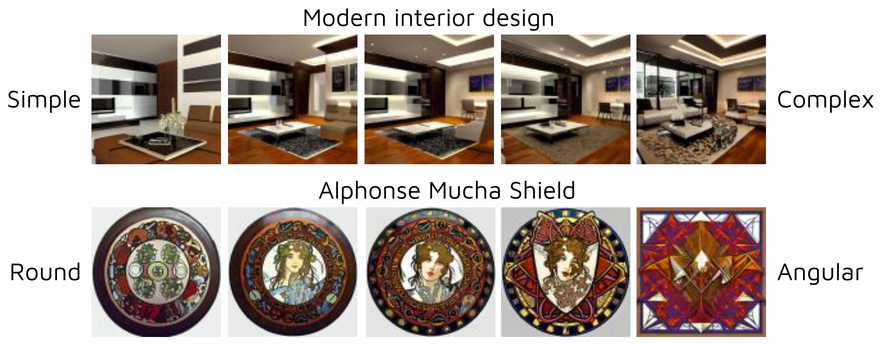


Figure 7.7: Example results of directional prompts. The center result is the result without a directional prompt and the left and the right are results of applying directional prompts. The rightmost and leftmost results applied the full vector difference between the two end prompts.

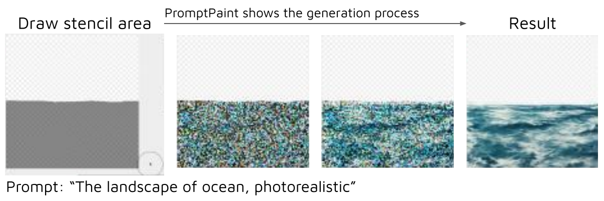

Draw stencil area
PromptPaint shows the generation process
Result
Prompt: "The landscape of ocean, photorealistic"

Figure 7.8: With a prompt stencil, the user can specify the area of generation (dark grey). When the user completes brushing the area, the tool starts generating a part of the image while showing the process to the user.

The user can add a new directional prompt with the + button, and for each of them, they can set two ends with prompts, which decide the direction of the attribute to add. Each end also has its own color, which the user can specify. After setting two ends, the user can move the position of the slider to decide the intensity of the attribute to add. For example, in Figure 7.1d, if the user wants to add a slight amount of the "matte" attribute, they can move the slider to the side of "matte." As the user moves the slider to one end in the directional prompt, the background color of the Prompt Palette gradually changes to the unique color of the end. When there are multiple directional prompts, this color changes to a mix of colors from the ends, with weights according to the slider values. Figure 7.7 is an example of adding other attributes with directional prompts.

7.4.2.4 Prompt Stencil

After setting the prompt to use, the user can gradually build the visual images with a prompt stencil or by specifying the generation area. The interaction is simple. While DRAW is selected (Figure 7.1e), the user can specify the area to generate on the canvas with brushing interaction (Figure 7.8). As the user completes brushing, *PromptPaint* starts to generate an image, with intermediate generation results shown to the user in real time. The progress bar in Figure 7.1g shows

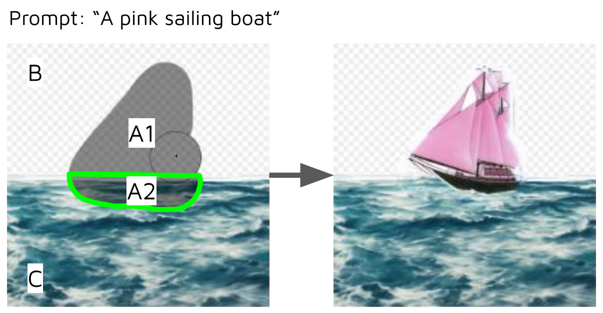


Figure 7.9: When applying prompt stencil upon the existing image, *PromptPaint* considers those existing images for generation. For the area where the stencil is overlapping with the existing image, *PromptPaint* generates a new image that is similar to the existing image based on the overcoat value (higher, less similar).

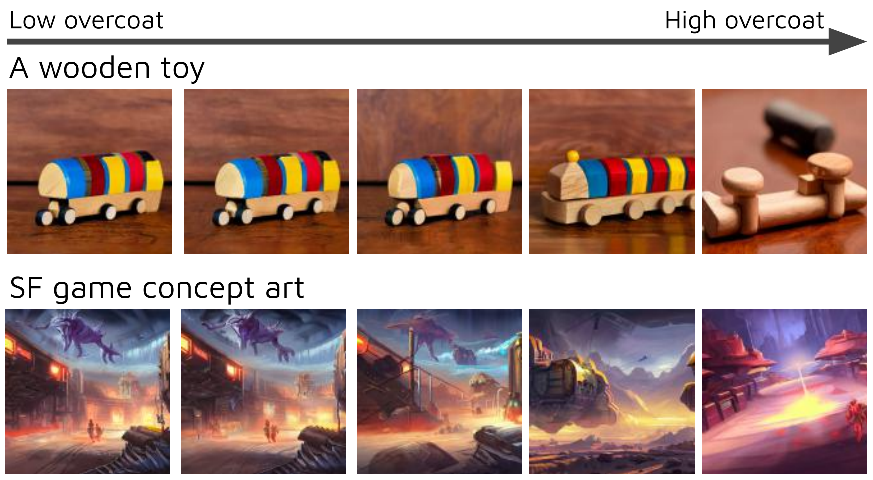


Figure 7.10: Example results of overcoating generation. In this figure, the image is generated again from the far left image with varying overcoat values (more right, higher overcoat values). Images generated with higher overcoat values tend to be less similar to the original image.

how much of the generation is done. Note that the user can adjust the intensity of the guidance and the number of intermediate steps with the sliders GUIDE SCALE and STEPS in Figure 7.1e. The user can repeat this process to fill in other canvas areas.

When the canvas already has images on the canvas layer, *PromptPaint* considers those already existing content to generate new content. For example, if the user has already generated the ocean scene and tries to place a boat on it, *PromptPaint* generates the boat considering the ocean scene (Figure 7.9). As in the green area of Figure 7.9A2, there can be cases where the user's new stencil would overlap with the already generated images. In such a case, based on the OVERCOAT value (Figure 7.1e), *PromptPaint* tries to generate a new image that is similar to the already generated images (the higher the overcoat value, the less similar). Figure 7.10 shows the cases of generating an image again with the same prompt with varying overcoat levels.

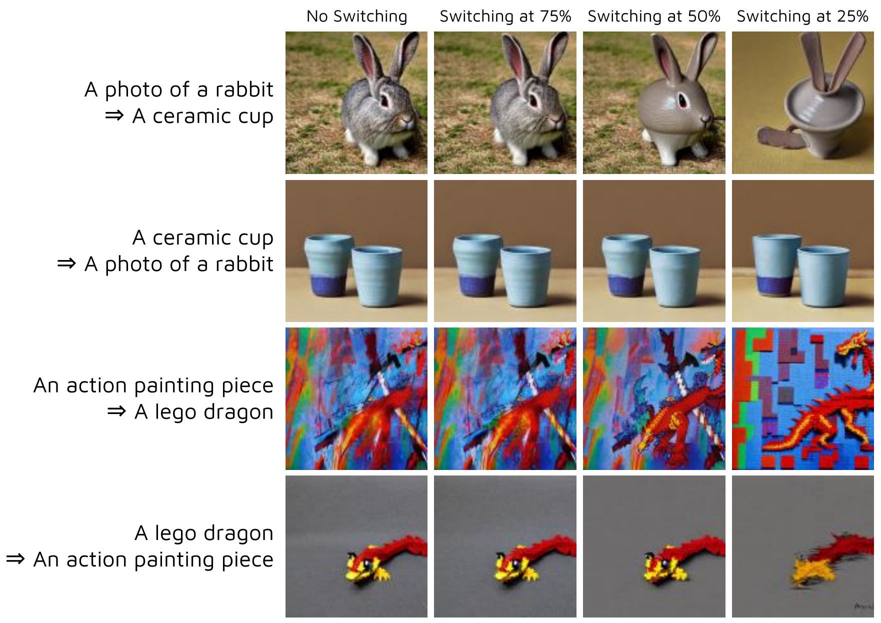

No Switching
Switching at 75%
Switching at 50%
Switching at 25%
A photo of a rabbit ⇒ A ceramic cup
A ceramic cup ⇒ A photo of a rabbit
An action painting piece ⇒ A lego dragon
A lego dragon ⇒ An action painting piece

Figure 7.11: Example results of prompt switching. Except for the first column which did not change the prompt during the generation (i.e., used the same prompt throughout the whole generation), each column switched the guiding prompt at a different point of the generation process.

7.4.2.5 Prompt Switching

PromptPaint allows users to interact with the generation pipeline even during generation, with *prompt switching*. With prompt switching, the user can change the prompt that guides the generation by changing either 1) the selection of the prompt in the Prompt Palette or 2) the slider values for directional prompts. With the change of prompts, as in Figure 7.11, prompts used in the earlier stage of the generation tend to decide the overall form and color, while those used in the later stage decide details. Hence, this technique can maintain the visual form of the generation with iterations. However, as in the second row of Figure 7.11, sometimes the generation result does not change much even with early prompt switching.

As the user can change the prompts during the generation, *PromptPaint* allows users to control the generation process. First, *PromptPaint* visualizes the prompts used in previous generations as paths in the Prompt Palette interface as in Figure 7.1c1. Note that the used directional prompts decide the colors of dots. This visualization of past paths helps users understand what they have tried, so that they can iterate more easily on different combinations of prompts. Furthermore, they can stop and restart the generation (the button in Figure 7.1g changes to either STOP or START). When the user has stopped generation, they can also roll back the generation to a certain step. It is possible either by selecting the past point in the progress bar in Figure 7.1g or undoing by hitting CTRL Z. If the user wants to switch to past versions of generations, they can click one of the dots on

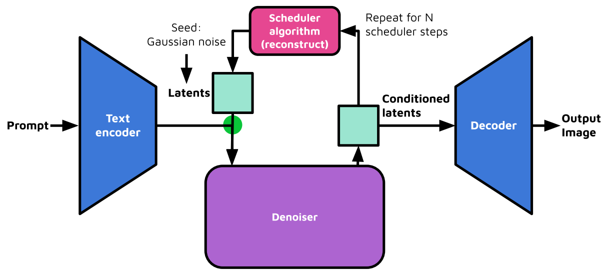


Figure 7.12: The pipeline of diffusion-based text-to-image models. The technical manipulation of *PromptPaint* happens in the green dot, either by manipulating vector-embedded prompts or intermediate latent.

the past paths. *PromptPaint* highlights the dot for the current generation step with a green border. When restarting the generation, the user can also set the number of steps that will be processed with a single "round," which is a single firing of generation. It is specifiable with SINGLE STROKE in Figure 7.1e. *PromptPaint* also allows users to change the area of generation during generation. When the user wants to exclude a certain area of generation from the current generation, they can use ERASE in Figure 7.1e. If they want to rerun the generation only on a certain part of the current generation, they can use REVISE in Figure 7.1e and specify the area.

7.5 *PromptPaint*: Technical Details

In this section, I describe how I technically enabled interactions of *PromptPaint*. First, I give an overview of diffusion-based T2I generation models. Then, I explain the technical approaches for each function and the implementation details.

7.5.1 Background: Diffusion-based Text-to-Image Generation Models

We focus on describing a high-level concept of how diffusion-based T2I models work in inference time (i.e., during generation). Diffusion-based text-to-image generation models [330, 343, 348] are composed of three parts (Figure 7.12): 1) text encoder that turns text prompts into vectors used to guide diffusion, 2) denoiser and scheduler, which gradually process image generation by reducing noise in the latent vectors of the image, and 3) decoder, which turns latent vectors into a higher resolution output image. Often the geometry of the latent representation corresponds to the output image, whereas the output image tends to have higher dimensions. Note that details of the architecture can vary between specific models, and we give more implementation details in a later section.

In the following sections, we introduce the technical approaches of *PromptPaint*. Note that all technical manipulations occur in the green dot of Figure 7.12, either by manipulating vector-encoded text prompts (prompt mixing, directional prompts, prompt switching) or latent from the denoiser and scheduler (prompt stencil).

7.5.2 Prompt Vector Manipulations

Prompt mixing and *directional prompt* manipulate the prompts in the vector space embedded by the text encoder. Prompt mixing simply interpolates different prompt vectors with weights [330] from the user's Prompt Palette:

$$v_{p_m} = \sum^{N} w_{p_i} v_{p_i} \tag{7.1}$$

In the above equation, v_{p_m}, w_{p_i}, and v_{p_i} represent the interpolated vector, the weight of each prompt, and the embedded vector of each prompt. On the other hand, the directional prompt first calculates the directional vector of two different concepts as follows:

$$v_{d_j} = v_{d_j 1} - v_{d_j 2} \tag{7.2}$$

where v_{d_j} indicates the directional vector and $v_{d_j 1}$ and $v_{d_j 2}$ stand for the embedding of the text of the prompts at both ends. Then, with weights on the directional vectors of the slider interfaces, *PromptPaint* calculates the final input of the prompt vector as follows:

$$v_f = v_{p_m} + \sum^{M} w_{d_j} v_{d_j} \tag{7.3}$$

Then v_f is used as input to the denoiser model to guide the denoising process. Users can change v_f during generation to accumulate and use different prompts for different parts of the generation process, which is how *prompt switching* is technically done.

7.5.3 Latent Manipulation

Prompt stencil and overcoated generation require manipulation in latent representations before being processed by the denoiser. Specifically, *PromptPaint* manipulates latent representations considering different areas with or without stencils or existing image content. This approach is similar to image-to-image diffusion in previous work [20]. For the area that is stenciled but does not have any image content (Figure 7.10A1), no extra latent manipulation is performed. However, for the stenciled area with image content (Figure 7.10A2), latent representation is manipulated as follows:

$$l_{m,k}(x,y) = \begin{cases} \nu(l_I(x,y),k), & \text{if } k < (1 - o/100) * K. \\ l_k(x,y), & \text{if } k >= (1 - o/100) * K. \end{cases} \tag{7.4}$$

As in the above equation, when the diffusion step (k) is smaller than the threshold $(1-o/100)*K$ (where o is the overcoat ratio and K is the number of all diffusion steps), $l_{m,k}(x,y)$ (the latent representation after manipulation at the position of (x,y) and the step of k) is replaced by $\nu(l_I(x,y),k)$ (where l_I is the latent representation of the existing image and $\nu(l,k)$ is a function that adds noise to the latent representation to the amount adequate to step k). Otherwise, the latent representation would not be manipulated and *PromptPaint* would use the reconstructed latent representation from the scheduler algorithm. For unstenciled areas, the area of Figure 7.10B is considered to have a white background. Then, the area of Figure 7.10B and C would be replaced as follows:

$$l_{m,k}(x,y) = \nu(l_I(x,y),k) \tag{7.5}$$

Therefore, these unstenciled areas are replaced by the latent representation of existing images with noise added for each step.

7.5.4 Implementation

I implemented *PromptPaint* as a web application using HTML, CSS, JavaScript, and React. For deploying a diffusion-based T2I model, I implemented a WebSocket-based Flask server, as *PromptPaint* needs to show intermediate generation results in real-time. For the diffusion model, I used Stable Diffusion [343], which uses UNet for the denoiser and the variational autoencoder decoder for the decoder. Specifically, for the UNet and the variational autoencoder models, I used `runwayml/stable-diffusion-v1-5`[4]. For the text encoder, I used the `openai/clip-vit-large-patch14` checkpoint[5] of CLIP [327].

To allow users to observe the intermediate generation process of diffusion models, *PromptPaint* decodes conditioned latent for every step. It puts more computational load on the pipeline, hence I optimized the use of diffusion models with half-precision models.

[4]https://huggingface.co/runwayml/stable-diffusion-v1-5

[5]https://huggingface.co/openai/clip-vit-large-patch14

Table 7.1: Participants in the user study. Their expertise in visual arts and the domain of interest are shown. We also show their experience using T2I models.

	Visual art	Year	Domain	T2I		Visual art	Year	Domain	T2I
P1	Hobbyist	5	Vector arts	Yes	P5	Novice	N/A	N/A	No
P2	Hobbyist	10	Paintings, cartoons, graphic arts	No	P6	Novice	N/A	N/A	Yes
P3	Hobbyist	20	Sketches, paintings	No	P7	Hobbyist	3	Sketches	No
P4	Hobbyist	30	Simple drawings, paintings	Yes	P8	Novice	N/A	N/A	No

7.6 User Study

I conducted a user study to understand how *PromptPaint* extends the use of diffusion-based T2I models by adopting the interactions that we have with paint mediums. Specifically, I focus on how interactions of *PromptPaint* could affect the user's experience in the exploration and steering of T2I generations to "create" visual artifacts. Therefore, I conducted an observational study with qualitative analysis.

7.6.1 Participants

I recruited eight participants (five female and three male, ages 22-51, M=28, SD=9.53) through university mailing lists. I asked participants to do a prescreening survey, checking if they can participate, as the study requires participants to see and hear. The prescreening survey also asked about the experience of participants in the visual arts. I recruited hobbyists or novices, as experts would be less likely to use automated generation tools in their practice (i.e., they have the expertise to create visual arts by themselves). Three of the participants had experience using T2I models. Table 7.1 shows the details of the participants. I gave each participant an Amazon gift card worth $20.

7.6.2 Procedure

I conducted an in-person lab study. I first asked participants to complete a pre-survey, which asks about their previous experience in visual arts (e.g., specific domains they have experienced, if they have) and using T2I models. The pre-survey also asks for demographic information. Then I showed the participants a video with an overview of the study (5 minutes). As we asked participants to think aloud during the study, this video instructed participants about the concept and an example of think-aloud. The overview video also lets participants know the basic functions of *PromptPaint*, which are raster image editing functions other than image generation (e.g., brushing, erasing). After instructing those functions, the video explained to participants how to generate images with

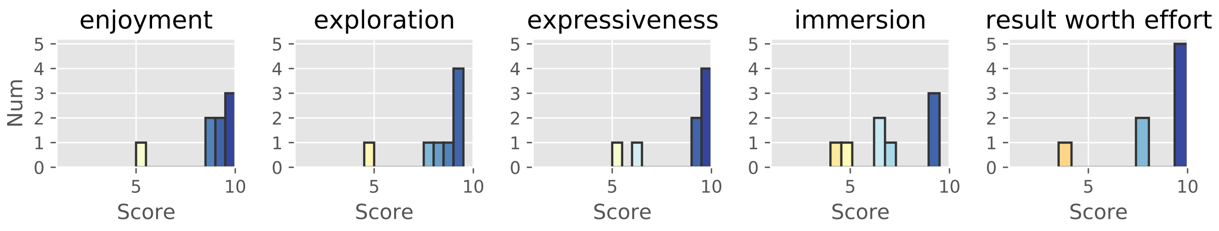


Figure 7.13: The histogram of how participants answered the creativity support index questions. The high score indicates that the participant perceived that *PromptPaint* supports the criteria.

a single prompt. After the first video, I asked the participants to try the functions in *PromptPaint*. The participants then went through four rounds of task sessions for four functions: prompt mixing, directional prompt, prompt switching, and prompt stencil. The participants went through sessions in the order of prompt mixing, directional prompt, prompt switching, and prompt stencil, since the latter functions require knowledge of the previous ones. For each task session, participants went through four steps: 1) watching an instruction video, 2) trying out the function as a tutorial, 3) conducting a task of creating visual artifacts as they want, while doing think-aloud, and 4) completing a post-task survey. Each instruction video took 1-2 minutes. Each tutorial took about 5 minutes. During the task, I asked the participants to actively try the function that they have just learned within 10 minutes. Post-task surveys asked participants if the function they had just tried facilitated 1) exploration of good surprises or 2) control of image generation. After all functions, participants were asked to complete an exit survey, asking questions about the general usage of the tool. This survey asked questions in the creativity support index [59], except those that questioned whether the tool helped collaboration. The post-survey also asked about the participant's sense of ownership and contribution, and if they felt they were collaborating with the system. After the post-survey, we conducted a short interview with the participants. In the interview, we asked about their strategies for using *PromptPaint*, how they felt about the ownership of the artifacts, and their impression of the four functions of the tool. The entire study took no more than 100 minutes.

7.6.3 Result

I report on the results of surveys, observations of the task with think-aloud and interviews.

7.6.3.1 Survey results

Figure 7.13 shows the responses of participants to the creativity support indexes. Participants were generally positive about *PromptPaint*, perceiving that it facilitated enjoyment, exploration, expressiveness, and immersion, while the results were worth their effort. However, there was one

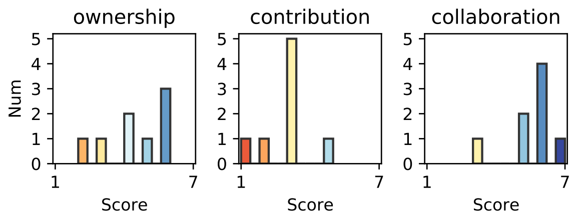


Figure 7.14: The histogram of how participants answered the question about the sense of ownership, contribution, and collaboration. The higher the scores, the participant felt that they have more ownership than *PromptPaint*, they contributed more than AI, and they collaborated with AI functions.

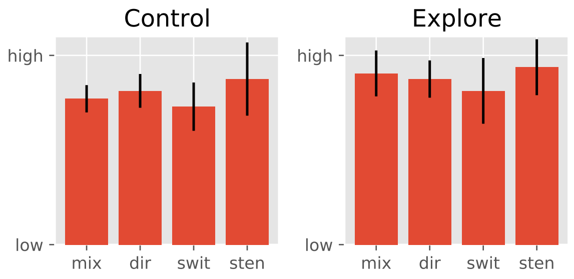


Figure 7.15: Comparison of four different functions on if they helped with 1) controlling or 2) exploring generation. `mix`, `dir`, `swit`, and `sten` stand for *prompt mixing*, *directional prompt*, *prompt switching*, and *stencil*, respectively.

participant who responded neutrally or negatively to these questions. It was the case where the participant had slightly more concrete expectations of what to create. Moreover, for immersion, there was one participant who answered negatively about the immersive aspect of the tool.

Figure 7.14 shows how participants felt ownership of the generated images, how much contribution they made (compared to AI), and whether they felt that they collaborated with *PromptPaint* in creating the artifact. Interestingly, participants felt that AI contributed more, but many still answered that they have some ownership of the generated artifact. At the same time, the participants tend to answer that they collaborated with *PromptPaint* at some level.

Figure 7.15 shows the participants' perceptions on how each function supported 1) the control of generation and 2) the exploration of interesting and good surprises. Overall, participants perceived all functions positively. While it is difficult to learn significant differences between functions due to the small size of the data, participants tend to perceive that the prompt stencil helped the most with controlling and exploring generation, while switching prompts helped the least.

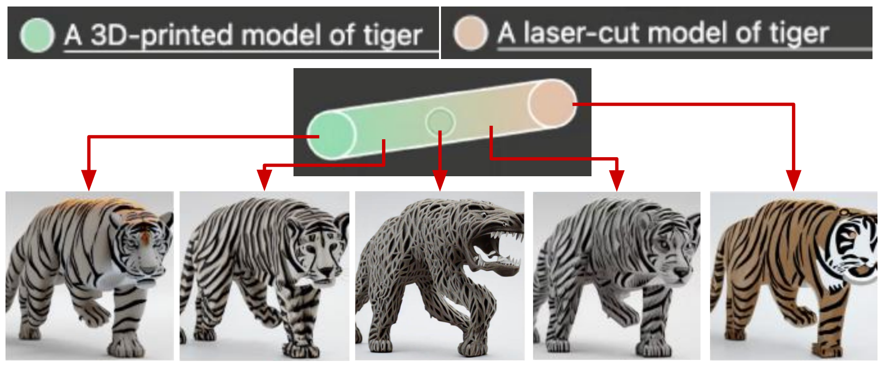


Figure 7.16: Prompt mixing from P5. By mixing semantically close prompts, the user can control the generation to explore the image space in between.

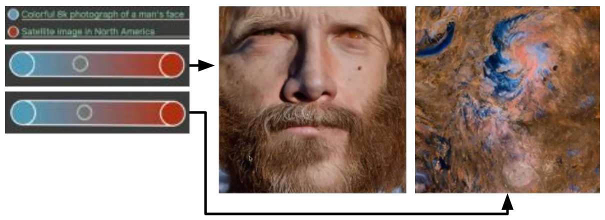

Colorful 8k photograph of a man's face
Satellite image in North America

Figure 7.17: Prompt mixing from P1. For semantically far prompts, the mixing prompts linearly in the vector space sometimes did not result in a linear impact on the generated images.

7.6.3.2 Qualitative Results

For qualitative results, I analyzed think-aloud, screen recording, and interviews by iterative coding with inductive analysis. I present findings on four functions, trade-offs in designs, the complexity of AI, and ownership issues.

7.6.3.3 Four functions

Prompt Mixing As seen in Figure 7.16, prompt mixing helped control generation, often allowing participants to explore the image space that is difficult to describe verbally (N=7). Some participants mentioned that visualizing the prompts and their intermediate spaces helped them flexibly explore and manipulate the prompt semantics (N=2). One interesting thing that one of our participants (P1) discovered was that when two prompts are semantically far, mixing concepts does not change the image linearly, but more in drastic “steps” between. For example, in Figure 7.17, P1 tried to mix two prompts, “Colorful 8k photograph of a man’s face” and “Satellite image in North America.” Here, at a certain boundary, a small increase of weights on one image could drastically change the image, indicating that the mix of weights did not impact the result linearly. One participant (P5) mentioned that they would like to mix more than three prompts, which was not supported

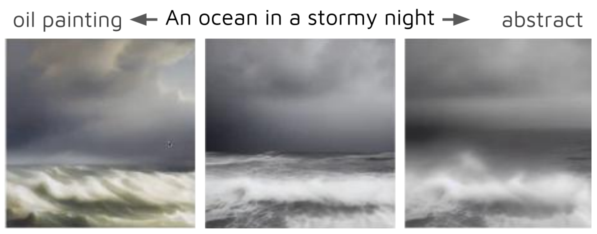


Figure 7.18: Directional prompt from P6.

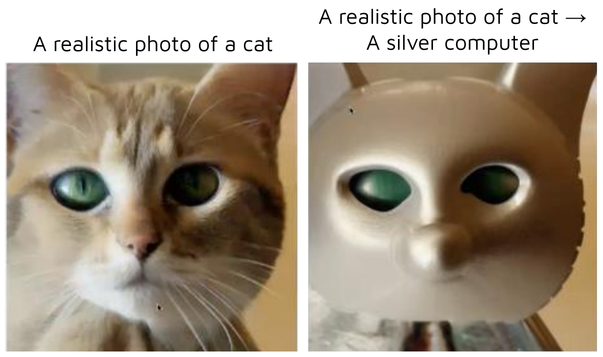

A realistic photo of a cat
A realistic photo of a cat → A silver computer

Figure 7.19: Prompt switching from P2.

in the current version of the tool.

Directional Prompt A directional prompt could help participants add attributes that do not exist within the original prompt they have tried (N=6, Figure 7.18). P3 also mentioned that the function helped to explore and perform fine-grained controls between two opposite concepts: *"I think the strength of this is like you can see how opposed things are, and how it is seen in between ... Basically this function allows you to explores something that is in between. Sometimes it's very difficult to like imagine things, and this would have been very helpful in that."* One of the challenges in using directional prompts was deciding on two opposite sides of the prompts (N=5). Sometimes, they struggle to bring up prompts that have semantically opposite meanings. In such cases, the participants expected that *PromptPaint* could have recommended the options for opposite prompts after the user inputs one prompt. In other cases, the way the participants interpret the prompts did not align with *PromptPaint*'s. We explain this further in Section 7.6.3.5.

Prompt Switching The participants thought that the switch of prompts can generate interesting mixes of prompts from what they observed from the examples shown in the tutorial and their sessions (N=6). P2, who created the image in Figure 7.19, mentioned: *"I think the strength of that is making something completely ridiculous and fun and changing an aspect of something to match something else."* However, prompt switching was the most difficult to use (N=5). Some partici-

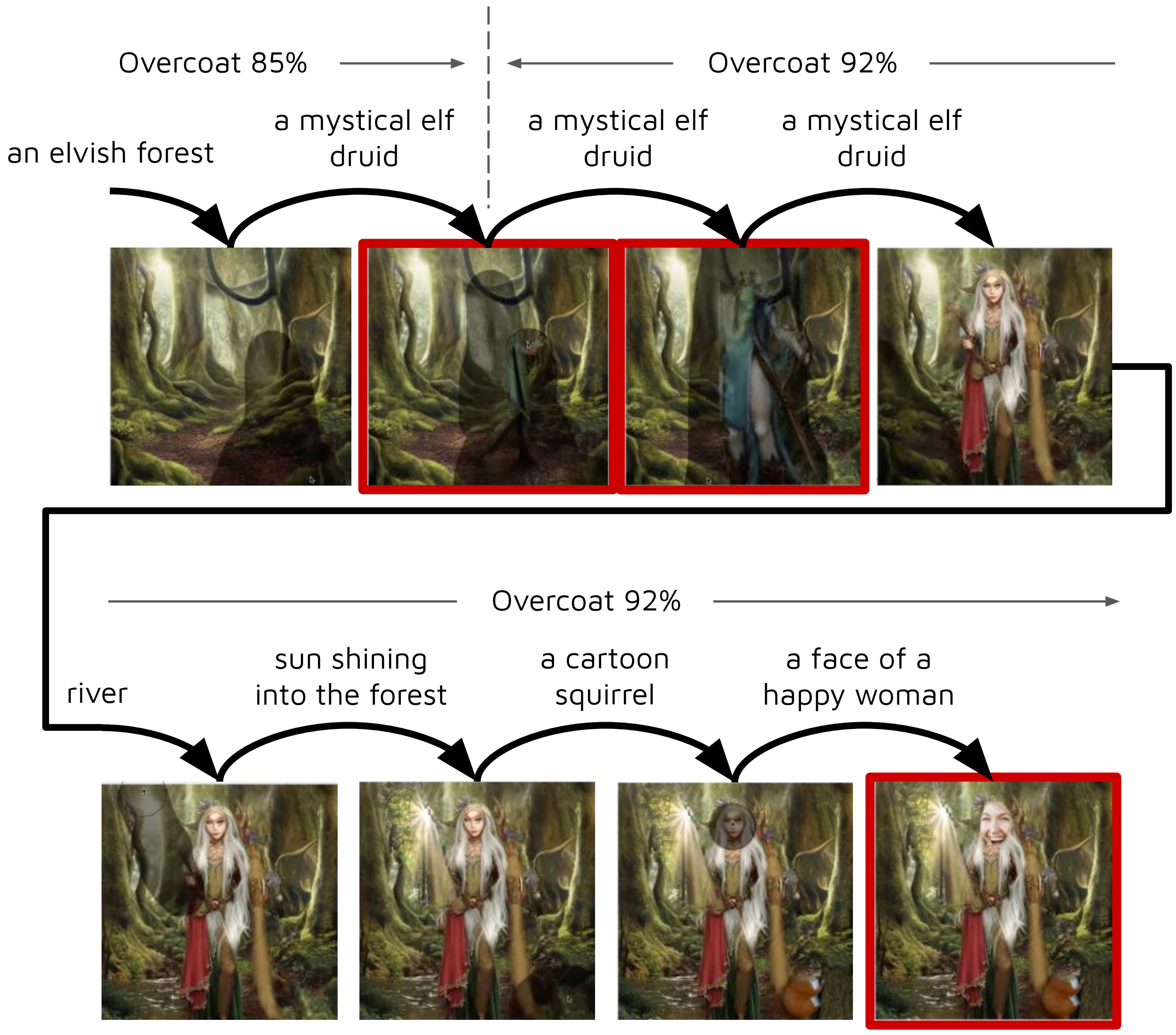


Figure 7.20: Prompt stencil from P2.

pants were unable to create a satisfactory image. It was due to difficulty in deciding when to switch prompts because it was hard to guess the result only by seeing intermediate generation results (i.e., noisy images during the diffusion process). The interface showing all previous generations (in the prompt palette in Figure 7.1c) could help users understand the previous generations they have tried (N=3) and iterate on the prompt switching. However, it was easy to clutter the interface with multiple rounds of iteration.

Prompt Stencil Participants mentioned that the prompt stencil allowed users to perform fine-grained controls with localized image generation (N=7). As in Figure 7.20, participants could gradually create the image by adding and changing visual elements in the scene. Participants could also adjust the overcoat level to generate partial images that are more or less similar to the existing ones. However, the prompt stencil also had limitations. For example, as in cases of "a mystical elf druid" in Figure 7.20 newly generated parts could be incomplete from the user's perspective. Furthermore, as in the case of "a face of a happy woman" of Figure 7.20, generated images could be mismatched with the existing ones. In some cases, the style of the newly generated images did not match the existing ones.

7.6.3.4 Design Trade-offs

Participants mentioned two potential trade-offs in the design of T2I generation tools. The first was the design trade-off between focusing on one image versus curating many results (N=2). I designed *PromptPaint* to allow users to iterate on a single canvas, giving users the experience closer to "gradually creating an image." However, due to stochasticity in the diffusion model (e.g., randomness from different seeds), participants found that seeing multiple results would be helpful in some cases. Furthermore, I designed *PromptPaint* to allow users to have more controls and interventions. For example, the tool allows users to have a high degree of freedom to change the prompts during generation. Although such designs open up new interactions in using diffusion-based T2I tools, some users found such designs too manual (N=2). Participants mentioned that the balance between automation and manual interventions would help.

7.6.3.5 High Complexity and Randomness of AI

Participants mentioned that the high complexity and randomness of AI behaviors were limitations of *PromptPaint* (N=7). Such complexity and randomness could make the generation result misaligned from the user's intention. Dissatisfaction tends to be more intense when the user has a more concrete picture of what they want to see from the generation results. For example, P7 thought that prompt stencils often failed to generate image parts with consistent perspectives and showed the greatest dissatisfaction with the prompt stencil. To facilitate generative AI even with such barriers, participants adopted some strategies. First, they tried to understand how AI works in simple settings (e.g., using a single prompt) and then applied more complex functions (e.g., prompt mixing) based on their understanding (N=2). Some users tried to understand how the model "interpret" prompts by using functions that interpolate or extrapolate the semantics of the prompts (N=2). For example, P1 interpolated the prompts of an apple and a pear to learn how the machine interprets the main attributes of each fruit. As participants also struggled to find prompts that work effectively for their intentions, they also actively used explanations and examples given during the tutorial (N=5).

7.6.3.6 Ownership and Contributions

Participants felt some ownership of the resulting visual images (N = 8), with degrees varying between participants. They mentioned that they contributed high-level ideas about the created artifact, while AI contributed low-level ideas and implementations. P1 mentioned that they felt like they became like "Steve Jobs" and AI would be "an Apple employee," and P2 thought that using *PromptPaint* felt like doing an art commission with more control power. A participant (P5) felt less ownership of the resulting piece because they were a novice in visual arts. P5 mentioned: *"I think AI contributes more than me and it's because I'm a novice. I did not paint at all, and I don't use any drawing software as well. So I think all the beautiful images are created by AI instead of me. I just specify the position, and it's just parameters."* Some participants also mentioned that they felt more ownership in the resulting artifacts if they align with what they expected (N=2). One participant mentioned the potential issues in the legal ownership of the generated artifacts, showing concerns about the copyright.

7.7 Discussion

Based on the suggested design of generative tools and the findings of *PromptPaint*, I discuss 1) the generalizability of paint-medium-like interactions in generative tools, 2) in-generation interactions for T2I models, 3) design trade-offs in generative tools, and 4) ownership issues.

7.7.1 Generalizing Paint-Medium-Like Interactions for Generative Tools

In *PromptPaint*, paint-medium-like interactions could facilitate effective, efficient, and iterative steering of T2I models. We can apply the idea of paint-medium-like interactions beyond *PromptPaint*. First, for the interaction of mixing colors, discrete colors can expand beyond prompts to other inputs, such as examples. Although interactions for color mixing would generally apply across different modalities, modularized generation specifications would need to be revised for each content modality. For example, for the generation of 3d models [188, 322], users would need to be able to select parts of 3d spaces to iterate. Similarly, for content that has sequential axes, such as text, video, or music, the generation specification would need to consider the sequential dimension [66]. In the interface, they can be instantiated in "sketches of different colors along the sequential axis." Specifically, for videos, which have both spatial and sequential dimensions, interactions for modularized specifications would likely be more complex than for other mediums. Still, the design pattern of applying different semantics (analogically, colors) to different parts of the artifact would generally hold across modalities.

For in-generation interventions, in *PromptPaint*, the prompts used in the earlier stage decided the overall form and colors, while the later ones decided on details. While I discuss how we can expand in-generation interventions in T2I models in the next section, here I talk about how we can apply in-generation interventions in other modalities. From the case of diffusion-based T2I models, we could observe that the earlier part of the gradual generation process decides "high-level characteristics," while the later stage decides "details." For other mediums also, models can be designed to gradually generate content in similar patterns. In such cases, training these models to follow human-understandable hierarchical structures would facilitate users to intervene during the generation process. For example, music generation algorithms can generate in the order of song structures, bars with chords, notes in each bar, and then embellishments such as legato or staccato.

7.7.2 In-generation Interaction for T2I models

A novel interaction in *PromptPaint* is that this tool allows users to interact with generative models *during the generation process* by changing the prompts. With diffusion models that gradually turn noisy images into clear ones, earlier prompts formed the overall composition, while the later prompts decided on details. While participants found this function interesting, they were also confused when adjusting their prompts during the generation. They struggle to find the right moment to change the prompt. For example, if diffusion models already processed many steps, changing the prompts did not significantly alter the outcome result. Seeing and interpreting intermediate representations might help to overcome such limitations. For example, if the intermediate noise-added image already has quite a concrete object, it might indicate that changing the prompt might not be effective. However, many users are unfamiliar with such noise-added images and do not know how to interpret them. Therefore, users would need to *learn* to interpret noisy images, which puts a greater load on the use of the tool fluently.

Making intermediate results more understandable to human users would be one approach to facilitate in-generation interactions for diffusion-based T2I models. As diffusion models do not necessarily need to generate an image from noise [23], we might be able to train diffusion models to follow other generation processes. For example, diffusion models that gradually concretize images from more pixelated ones would be more understandable to the users, allowing them to grasp what the model might generate from the current intermediate step. Moreover, such representations can allow users to edit the intermediate results. For example, with pixelated intermediate images, if the user is generating a human face and spots a "blonde" color in the area of hair, they would be able to change the color of the hair by changing the region to other colors. With such an interaction, the user would be able to flexibly steer AI generation.

7.7.3 Design Trade-Offs in Generative Tools

From the user study, I found design trade-offs for generative tools. First, while "creation tools" often assume a single artifact to be created (e.g., a single canvas for image editors), due to the complexity and randomness in generative models, generative tools would require some levels of "curation" of multiple results. Providing both features would allow users to have steering experiences while addressing some issues with the randomness of algorithms. For example, a generative tool can have multiple rounds of interactions that first receive user specifications, generate a set of candidates, allow users to select one of them, and then iterate. For effective curation, it would be valuable to learn the user's preferences during the interactions and provide options that would better align with the modeled user preferences. The second trade-off in design is the balance between automation and manual controls. For such a trade-off, having "low threshold and high ceilings" [336] would be helpful. That is, simple and automated interactions can be a "low threshold" way to steer the generation, while more manual steering interactions can be a "high ceiling" option.

7.7.4 Ownership of Generated Artifacts

From the user study, I found that the users had some sense of ownership of the artifacts generated, as they contributed high-level ideas. However, at the same time, as *PromptPaint* contributed ideas and implementations of lower levels, they felt less ownership than creating artifacts themselves. For generative creation tools to secure the user's sense of ownership regarding the final artifacts, it would be important to understand which aspects of artifact creation the user thinks are important. For example, for users who do not put a lot of value on manual labor, automating some parts of the artifact creation would not hurt the user's sense of ownership much. On the other hand, if the user values the skills and efforts involved in the creation of artifacts, then automation would hurt the sense of ownership. Hence, the tool would need to understand the user's values and allow users to select from which part they would like to get assistance from generative AI. While I designed *PromptPaint* to add user interactions to already generative AI models, to build practical tools, we need to embed generative functions into existing workflows so that we can preserve user values in their workflow.

Another issue of ownership is more legal, such as not hurting copyrights when using the tool. As existing diffusion models have been shown to copy content from the training dataset [376], it would be crucial to carefully curate the training dataset so that the generative models do not damage the legal ownership during their use. Although researchers have started to exclude images if the original owners do not want their images in the dataset[6], it is still opt-out. Moreover, there

[6]https://haveibeentrained.com/

could be some trade-offs between preserving legal ownership and having a large-scale dataset. Potentially, data augmentation can be a way to transform training datasets to balance the preservation of ownership and the scale of the data.

7.8 Conclusion

In this chapter, I demonstrate intuitive and iterative steering interactions for T2I generative models. I introduce the design approach of interacting with generative models in a way similar to how we interact with paint mediums. This design approach allows users to explore the semantic vector space in a way similar to how we mix colors and gradually build the artifact with different semantics in a way similar to how we apply colors to varying parts of the painting process and the canvas. The motivation is to make end-to-end usage of generation models more flexible and gradual with iterative steering. I apply the design approach in diffusion-based T2I models and introduce *PromptPaint*. *PromptPaint* adopts four steering approaches, prompt mixing, directional prompts, prompt switching, and prompt stencil. From the user study with *PromptPaint*, I identify how people use the suggested interactions and potential insights on how we should design and build future generative tools. This chapter links back to the design approach for steering interaction, as *PromptPaint* adopts paint-medium-like interactions to facilitate steerability in the use of T2I models.

CHAPTER 8

Reflections

In this section, I reflect on how the three systems, *Artinter*, *TaleBrush*, and *PromptPaint*, extend art-making experiences, either by broadening the way we create art or alleviating pain points in art-making. I also discuss how I applied the design approaches learned in Chapters 2 and 3 to three tools. Finally, I present potential future directions not yet addressed in my dissertation.

8.1 Designing Steering Interactions for AI-powered Art-making Tools

My first tool, *Artinter*, focused primarily on alleviating challenges in visual art commission by adding multi-modal steerable interactions based on the user's communication contexts. First, with the mood board interaction to facilitate collaborative communications, *Artinter* learns the concepts of users. Second, users can use sliders of user-defined concepts to search and generate more references. Note that slider interactions are easy to understand and use, even for novice users. Moreover, with concepts that AI learned, users do not have to retreat from their own mental models to use search and generation functions, which leads to efficient use of those functions. Sliders were used iteratively, allowing users to effectively communicate their ideas.

TaleBrush introduces a novel approach to the generation of story text by combining visual sketching inputs with textual inputs. With this design, *TaleBrush* facilitates effective, efficient, and iterative expressions of the user's intention. Sketching is what story writers already adopt to plan out their stories with the fluctuation of the character's fortune. It also alleviates the cumbersome specification of a series of inputs for story dynamics, as users can specify multiple fortune input values with a single stroke. Iteration in generation becomes easy, as the user can redraw the part of the sketch for the part they want to see newly generated texts.

PromptPaint provides a novel way of manipulating guiding prompts for image generation by combining it with interactions of paint mediums. It allows users to mix prompts in different ways and modularize generations so that they can gradually build the visual artifact. With expanded interactions, *PromptPaint* allows users to steer image generation that would be difficult with natural language prompts alone. These interactions can be more efficient than text prompts and also facilitate iteration by allowing users to make fine-grained edits to the guiding prompts.

Technically, all these tools translates steering input interactions into the manipulation of vectors in semantic space, then generated artifacts based on the manipulated vectors. For example, for *Artinter*, images on the mood board are first transformed into vector embeddings, then concepts are learned on the vector space with concept-building interactions. Then, search and generation functions are steered with slider interactions, which modify vectors with concept vectors. For *TaleBrush*, it translates the interpolation of vectors for discrete fortune control codes into sketching interactions, where the y-position of the sketch maps to the level of interpolation. *PromptPaint*, similarly, first transforms user prompts into vectors and then allows users to manipulate them with various paint-medium-like interactions.

In summary, the three systems introduce new ways to broaden art-making expressions and alleviate pain points with effective, efficient, and iterative steering interactions that mix familiar input modalities.

8.1.1 Reflection: Applying Design Approaches to Other Domains

When designing the steerability of AI-CSTs, the designer would need to consider the unique characteristics of each domain. First, to facilitate iteration, the AI-CST designer would need to design AI output that aligns with the intermediate representation that users are familiar with. Comprehensible intermediate representations would allow users to easily iterate on the creation, as they can flexibly intervene within the AI generation. For example, for visual art generation models, it would be favorable if users could easily understand and edit the intermediate AI results. *PromptPaint*'s prompt switching function has limitations with this perspective, as its intermediate generation results are noise-added images, which are difficult to understand and uneditable by the users.

Second, for input interactions, the AI-CST designer should consider which aspect would be controlled by the user. That is, the artist might want to control those aspects that would define the uniqueness of each artifact to explore the possible artifact space. For example, *TaleBrush* allowed users to steer story generation with the dynamics of the character's fortune, as it is one defining factor of each story. Moreover, for the design of intuitive expressions, the AI-CST designers would want to adopt the existing familiar interactions that the artists already follow. For example, *TaleBrush* adopted visual sketching interactions, since there have already been existing practices of sketching the character's fortune when planning the story. In summary, considering specific art-making contexts would facilitate the design of steering interactions for AI-CSTs that are familiar to users and effective in their practices.

8.2 Future Directions

I suggest future directions that extend steerable AI-powered art-making tools.

8.2.1 Steering for More Complex Mediums

One future direction is to extend effective, efficient, and iterative steering interactions to tools for more complex creation. For example, video generation algorithms [373, 407] are emerging, but we are not sure how we should interact with these models to create the artifact that we want. These models started to provide prompting as one way to interact, but with the high complexity of videos, prompting alone is unlikely to ever be enough. A user would like to control the placement of objects in the scene, their animation along the temporal dimension, the audio played along the video, and many other aspects. Therefore, allowing users to make specifications according to their intentions can easily become complex. It would be important to design steering interactions while considering the design lessons extracted in Chapter 4. Transforming complex specifications into efficient interactions would relieve the burden of interacting with these models. Facilitating iteration would also alleviate the complexity of creating these artifacts, as users would be able to make specifications gradually. As video itself is multi-modal, multi-modal inputs would likely facilitate the use of video generation. However, which modalities should be used and how they can be combined in a video generation tool remain a question.

8.2.2 Leveraging Artificial Intelligence Further in AI-CST

While I focused on facilitating steering interactions, there may be more research opportunities to make use of AI technologies. One is to allow these AI tools to learn what the user wants and adapt their behaviors. With such an approach, AI outputs can be better aligned with the user's expectations, making steering more effective and efficient. In this dissertation, *Artinter* adopted this approach to some extent by learning the user language and allowing them to use it as a steerable handle for the search and generation function. There may be more to this approach, such as adapting the AI output as the user shows preferences for some of the intermediate outputs.

Moreover, in my dissertation, tools were designed for settings where users allowed for some interesting surprises from AI output. Even *Artinter*, which is for visual art communication settings, users were open to inspiration while using the tool, as communication usually occurs along with some ideation. Future work can investigate those settings where the user would have more concrete expectations about what they want. It would be similar to subcontract-like relationships in human-human support relationships in Chapter2. With these tools, stronger steering would be required, with narrower specifications and less divergence. Designing such tools would allow users to leverage AI technologies in various settings.

8.2.3 Steerable AI for New Types of Artifacts

While the tools I designed broadened the way of producing artifacts, they did not necessarily extend the type of artifacts that we can create. Historically, technological advances have contributed to the development of novel artifacts. For instance, photographs and movies could not have been art forms without the technological advances that enabled their creation. One question with AI technologies is what kind of novel artifact can we create with them. Current AI technologies focus on reproducing existing artifact types by using them as training data, and hence have been more bounded to existing artifact types in terms of output. How can we combine AI technologies to allow artists to create things that have never existed before? How can tool designers build steerable tools that can support such creations for a broader set of users? How should we train AI/ML pipelines to create those novel artifacts in high quality with steerable handles? All these questions will open up novel approaches to the use of AI technologies for new art-making.

8.3 Conclusion

In this dissertation, I investigate ways to use AI models to build usable and useful tools for art-making by adding steering interactions. AI/ML technologies introduce novel opportunities to broaden our art-making practices while alleviating challenges in them. However, they are not yet malleable and shapable enough for users to express their artistic intentions. I strive to combine steering interactions with these models so that AI/ML models can be flexible art-making materials. First, I study user needs and current designs of art-making tools to extract design approaches for steerable AI-powered art-making tools. For the study in Chapter 2, I identified what users would expect from intelligent types of support by interviewing artists about how they work together with *already-intelligent agents,* other people. In the study of Chapter 3, I reviewed previous efforts in designing art-making tools and found their design patterns that intersect used technologies, supporting roles, interactions, and intended users. Based on these, in Chapter 4, I propose the design approach of mixing familiar input modalities for effective, efficient, and iterative steering of AI-powered art-making tools. I demonstrated the design approach with three tools. The first is *Artinter*, which facilitates the communication of visual art commissions by combining mood board and slider interactions when using adaptive search and generation functions. The second is *Tale-Brush*, which adopts visual sketching interactions to steer the behavior of story generation models in the context of story co-creation of humans and AI. The third is *PromptPaint*, which extends prompt-based image generation by allowing users to manipulate guiding prompts with interactions similar to what they would do with paint mediums. Through these prototype systems, I demonstrate the effectiveness of our guidelines for building steerable AI-powered art-making tools. I hope researchers and tool designers can use the learned lessons for designing a broader set of new art-making tools.

APPENDIX A

Appendix for Chapter 4

A.1 Technical Evaluation

I technically evaluated *Artinter* in two ways: 1) how accurately the ML system recognizes artistic concepts with a small number of training data instances and 2) if *Artinter*'s concept-based search aligns with how human users perceive these concepts.

A.1.1 Evaluation of Concept Recognition

To evaluate how *Artinter* recognizes concepts, I conducted a simulated study. In the study, I ran *Artinter*'s concept learning pipeline to train different artists' styles as concepts. Hence, I considered each artist as a concept and their works as artifacts. I specifically used artists in the WikiArt dataset [419]. The focus was to identify the performance impact of the number of examples entered and the number of related concepts chosen.

First, among artists who have more than 50 artworks, I randomly sample 50 artists. For each artist, I sampled 35 art images as training data and 15 others as test data. With this sampled dataset of 50 artists, for each number of concepts trained (one to three), I randomly took 50 combinations of concept groups. For example, with two concepts related, I had 50 combinations of two artists. For each concept group, from each artist's training dataset, I sampled artworks to use as training data. I varied the total number of training artworks from 2 to 30. When multiple related concepts are trained, I sampled an equal number of artworks from each concept, so that the sum of them can be the total number of works to be provided. For example, if eight images were to be provided for two related concepts, I picked four from each concept. For each number of artworks used for training, I sampled training artworks 5 times, and for each sampled training artwork, I ran training five times, using a linear classifier. Then, I evaluated each classifier against the test dataset for each concept group. Only when one concept is trained, I used 35 images randomly sampled from the whole dataset as negative class samples. Among them, 20 were used as training data and 15 were used as test data.

A.1.1.1 Results

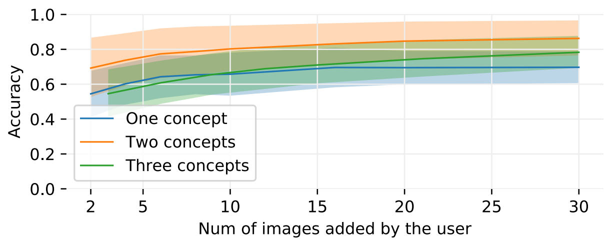


Figure A.1: Accuracy of recognizing concepts with the varying number of artifact instances and trained concepts. Shaded areas indicate the range of one standard deviation in accuracy.

Figure A.1 shows the result. As might be expected, accuracy increased with the addition of more data. I also found that training with two related concepts had higher accuracy than having one concept. Having three related concepts showed a similar performance to training a single concept, but only when the number of provided images is low. With two related concepts and at least 10 data points, *Artinter* showed recognition accuracy of 80%. This might be a reasonable performance, considering the small number of examples provided and that the art styles can be ambiguous and subjective [125]. Moreover, our evaluation approach trains on random artists' styles, where one artist's visual style might vary a lot, possibly giving some penalties to the results.

A.1.2 Evaluation of Search with Concepts

I evaluated search functionality to see if the human perception of concepts aligned with search results. In this analysis, I first trained a concept on an artist's artworks and performed a search with the trained concepts. With the query image and the top-ranked searched image, I asked crowd workers which image shared more style-wise characteristics with the images used to train the concept. If their perception of concepts aligns well with *Artinter*, they would find that the searched image is closer to the training images. I also asked their reasoning behind the answer. To search images that were likely to have the concept, I queried for the vector that added twice of CAV to the vector embedding of the query image. For images, I only used Abstract/Expressionism arts. Other artworks tend to have objects, which can confuse people when they answer our task (i.e., they would focus on content rather than style). Moreover, I assumed that users would likely have a more similar set of art in mind when they define concepts for the search function. To simulate this, I first randomly sampled a piece from an artist, and then, from the same artist, we additionally sampled pieces that are the most similar to the already sampled one. I also choose query images as those near the center of the vector representation space of the artwork dataset. This is because search results are less likely to reflect the concept if query images are at the edge of vector representation.

For example, it would be difficult to search for an image that is darker than an already extremely dark query image.

I conducted the experiment when one or two related concepts are trained. Similar to the evaluation of concept recognition, I first sampled 50 artists, and for each number of concepts, I sampled 50 combinations of concept groups. With one concept, I sampled six images to train the concept. For negative classes, I randomly sampled 20 artworks from the whole artwork dataset. With two related concepts, I sampled three images for each concept. I picked these numbers as they would be reasonably small for users to provide while assuring a fair performance of the classifier. For each concept group, I simulated the search once, resulting in 50 search results for each number of trained concepts. For each search query result, I asked the question to five crowd workers. In the two related concepts condition, I only showed training images from the concept that is used for the search. Workers were recruited from Amazon Mechanical Turk, who are in the US, have 99% of acceptance rate, and have been accepted for more than 1000 tasks. They were paid $0.25 for each task (about $10/hr payment rate).

A.1.2.1 Results

With one concept, the accuracy in estimating searched images was 64.0%, and with two related concepts, the accuracy was 77.2%. As implied by previous work [125], I found that there can be ambiguity in making the decision, as people sometimes focus on different style-wise elements (e.g., colors vs. patterns). It also means that some workers could see less similar style aspects between the searched image and example images. This signals that the concept-based search can provide "serendipitous" images, which can support inspiration [34, 102, 114]. Considering ambiguity, serendipity, and the small number of data instances used in training, the search performance can be at a usable level, specifically when two concepts are trained. However, as mentioned in the main studies, best practice was not often achieved for contextualization.

A.2 Examples: Impact of Relating Concepts

Figure A.2 shows cases of how relating concepts can impact the recognition and search results. In the example cases, I could observe that *Artinter* more accurately recognizes concepts and draws more relevant search results when concepts are related to each other. Specifically, for Figure A.2a, relating concepts facilitated *Artinter* to more accurately recognize and search for *blue*. In Figure A.2b, only with relating concepts, *Artinter* could search for examples that are more relevant to each concept. Figure A.2b, however, also shows limitations of training concepts with few examples. With concept-relating, searching for *clarity* leads to results with less of red colors, potentially

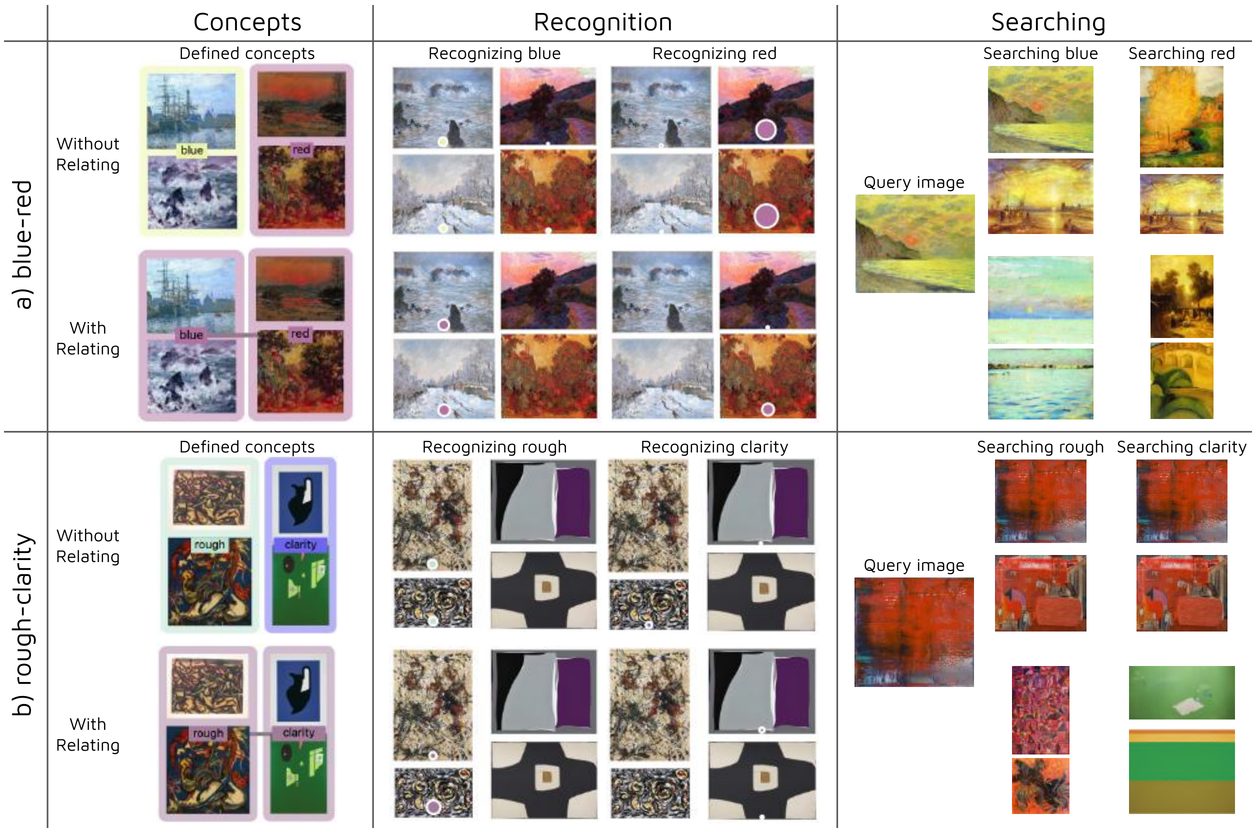


Figure A.2: Two cases of how relating relevant concepts impact the performance of recognition and search, for *blue-red* and *rough-clarity* concepts. For recognition, if there is a circle in the image, it means that *Artinter* recognizes the concept from the image. Bigger the circle is, more confident *Artinter* in recognizing the concept. For the search, I added the CAV of each concept to the vector representation of the query image and drew the nearest neighbors from the WikiArt dataset.

as *Artinter* learned that *clarity* is somewhat relevant to having green colors.

A.3 Examples: Impact of Weights and Scale in Generation

In Figure A.3, I demonstrate two cases generating images with varying weights and scales. As in demonstrations, while the user can control weights and scales separately, the result would be decided with the combination of these parameters.

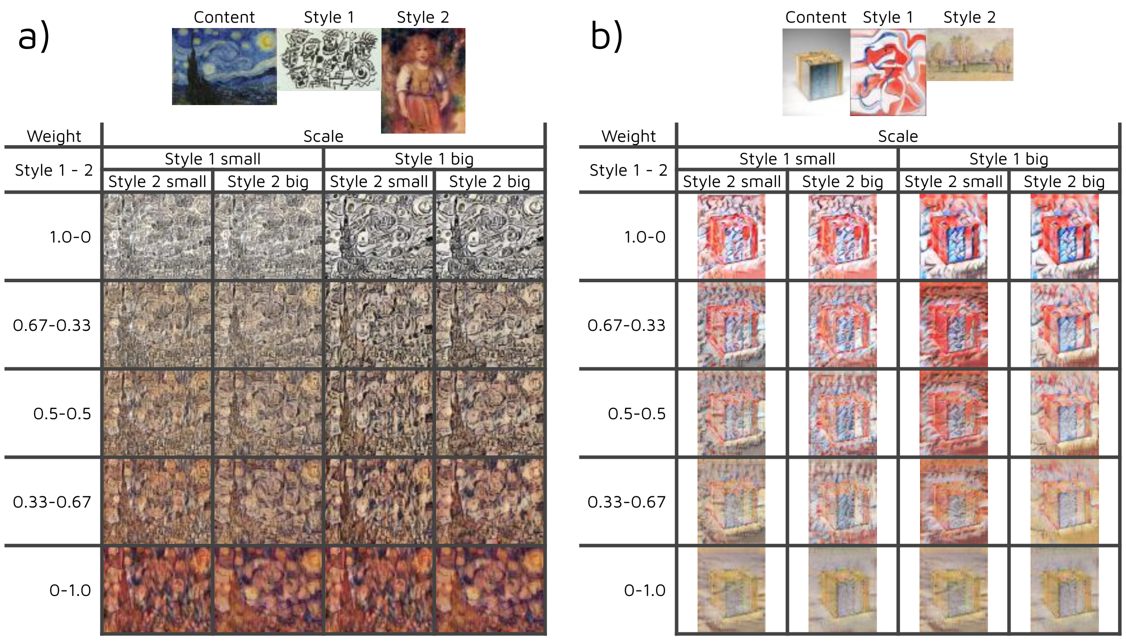


Figure A.3: Two cases of how varying weights and scales impacts generation. The scale of “small” has the half of the size presented in “Content,” “Style 1,” and “Style2.”

BIBLIOGRAPHY

[1] Rinat Abdrashitov, Fanny Chevalier, and Karan Singh. Interactive exploration and refinement of facial expression using manifold learning. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 778–790, New York, NY, USA, 2020. Association for Computing Machinery.

[2] Ai-Da. Ai-da robot, 2021. Accessed: January, 2021.

[3] Wolfgang Aigner, Silvia Miksch, Heidrun Schumann, and Christian Tominski. *Visualization of Time-Oriented Data*. Springer Publishing Company, Incorporated, 1st edition, 2011.

[4] Taketo Akama. Controlling symbolic music generation based on concept learning from domain knowledge. In Arthur Flexer, Geoffroy Peeters, Julián Urbano, and Anja Volk, editors, *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*, pages 816–823, 2019.

[5] Nader Akoury, Shufan Wang, Josh Whiting, Stephen Hood, Nanyun Peng, and Mohit Iyyer. STORIUM: A Dataset and Evaluation Platform for Machine-in-the-Loop Story Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6470–6484, Online, November 2020. Association for Computational Linguistics.

[6] Yuval Alaluf, Omer Tov, Ron Mokady, Rinon Gal, and Amit H. Bermano. Hyperstyle: Stylegan inversion with hypernetworks for real image editing. *CoRR*, abs/2111.15666, 2021.

[7] Safinah Ali, Tyler Moroso, and Cynthia Breazeal. Can children learn creativity from a social robot? In *Proceedings of the 2019 on Creativity and Cognition*, C&C '19, page 359–368, New York, NY, USA, 2019. Association for Computing Machinery.

[8] Teresa M Amabile. The social psychology of creativity: A componential conceptualization. *Journal of personality and social psychology*, 45(2):357, 1983.

[9] Teresa M Amabile. Componential theory of creativity. 2012.

[10] Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. Power to the people: The role of humans in interactive machine learning. *AI Magazine*, 35(4):105–120, Dec. 2014.

[11] Saleema Amershi, James Fogarty, and Daniel Weld. Regroup: Interactive machine learning for on-demand group creation in social networks. In *Proceedings of the SIGCHI Conference*

on Human Factors in Computing Systems, CHI '12, page 21–30, New York, NY, USA, 2012. Association for Computing Machinery.

[12] Saleema Amershi and Meredith Ringel Morris. Cosearch: A system for co-located collaborative web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, page 1647–1656, New York, NY, USA, 2008. Association for Computing Machinery.

[13] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N. Bennett, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, and Eric Horvitz. Guidelines for human-ai interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA, 2019. Association for Computing Machinery.

[14] Prithviraj Ammanabrolu, William Broniec, Alex Mueller, Jeremy Paul, and Mark Riedl. Toward automated quest generation in text-adventure games. In *Proceedings of the 4th Workshop on Computational Creativity in Language Generation*, pages 1–12, Tokyo, Japan, 29 October–3 November 2019. Association for Computational Linguistics.

[15] Prithviraj Ammanabrolu, Wesley Cheung, William Broniec, and Mark O. Riedl. Automated storytelling via causal, commonsense plot ordering. *CoRR*, abs/2009.00829, 2020.

[16] Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara Martin, and Mark Riedl. Guided neural language generation for automated storytelling. In *Proceedings of the Second Workshop on Storytelling*, pages 46–55, Florence, Italy, August 2019. Association for Computational Linguistics.

[17] Christopher Andrews. An embodied approach to ai art collaboration. In *Proceedings of the 2019 on Creativity and Cognition*, C&C '19, page 156–162, New York, NY, USA, 2019. Association for Computing Machinery.

[18] Cecilia R. Aragon and Alison Williams. Collaborative creativity: A complex systems model with distributed affect. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, page 1875–1884, New York, NY, USA, 2011. Association for Computing Machinery.

[19] Ilhan Aslan, Katharina Weitz, Ruben Schlagowski, Simon Flutura, Susana Garcia Valesco, Marius Pfeil, and Elisabeth André. Creativity support and multimodal pen-based interaction. In *2019 International Conference on Multimodal Interaction*, ICMI '19, page 135–144, New York, NY, USA, 2019. Association for Computing Machinery.

[20] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18187–18197, 2022.

[21] Thomas Ball, Shannon Kao, Richard Knoll, and Daryl Zuniga. Tilecode: Creation of video games on gaming handhelds. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 1182–1193, New York, NY, USA, 2020. Association for Computing Machinery.

[22] Kitti Balogh, Krisztina Szucs, Viktoria Verecze, and Zoltan Varju. Visualizing star wars movie scripts.

[23] Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie S. Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise, 2022.

[24] Howard S. Becker. Art as collective action. *American Sociological Review*, 39(6):767–776, 1974.

[25] Howard S. Becker. *Art Worlds*. University of California Press, 1984.

[26] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[27] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA, 2021. Association for Computing Machinery.

[28] Luca Benedetti, Holger Winnemöller, Massimiliano Corsini, and Roberto Scopigno. Painting with bob: Assisted creativity for novices. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, page 419–428, New York, NY, USA, 2014. Association for Computing Machinery.

[29] Walter Benjamin. The work of art in the age of mechanical reproduction. *Visual Culture: Experiences in Visual Culture*, pages 144–137, 1936.

[30] Eden Bensaid, Mauro Martino, Benjamin Hoover, Jacob Andreas, and Hendrik Strobelt. Fairytailor: A multimodal generative framework for storytelling, 2021.

[31] Guillermo Bernal, Lily Zhou, Erica Yuen, and Pattie Maes. Paper dreams: Real-time human and machine collaboration for visual story development. In *XXII Generative Art Conference*, Rome, Italy, 2019. Domus Argenia.

[32] Michael S. Bernstein, Greg Little, Robert C. Miller, Björn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. *Soylent: A Word Processor with a Crowd Inside*, page 313–322. Association for Computing Machinery, New York, NY, USA, 2010.

[33] Natalia Bilenko. *The narrative explorer*. PhD thesis, Master's thesis, EECS Department, University of California, Berkeley, 2016.

[34] Lennart Björneborn. Three key affordances for serendipity. *Journal of Documentation*, 2017.

[35] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large scale autoregressive language modeling with mesh-tensorflow, 2021.

[36] Kyle Booten and Katy Ilonka Gero. Poetry machines: Eliciting designs for interactive writing tools from poets. In *Creativity and Cognition*, pages 1–5, 2021.

[37] Mike Bostock. Les misérables co-occurrence, Apr 2012.

[38] Nadia Boukhelifa, Anastasia Bezerianos, Tobias Isenberg, and Jean-Daniel Fekete. Evaluating sketchiness as a visual variable for the depiction of qualitative uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2769–2778, 2012.

[39] Richard Brath. Surveying wonderland for many more literature visualization techniques. *CoRR*, abs/2110.08584, 2021.

[40] R. Brecknock. *A New Renaissance: Contemporary Art Commissioning*. Rosenthal Publishing, 1996.

[41] Matthew Brehmer, Bongshin Lee, Benjamin Bach, Nathalie Henry Riche, and Tamara Munzner. Timelines revisited: A design space and considerations for expressive storytelling. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 23:2151–2164, 2017.

[42] Write Brothers. Dramatica®the next chapter in story development, 1994.

[43] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer. An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 18(1):31–40, 1992.

[44] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[45] Jill Burstein, Beata Beigman Klebanov, Norbert Elliot, and Hillary Molloy. A left turn: Automated feedback and activity generation for student writers. In *Language Teaching, Learning and Technology*, pages 6–13, 2016.

[46] Daniel Buschek, Martin Zürn, and Malin Eiband. The impact of multiple parallel phrase suggestions on email input and composition behaviour of native and non-native english writers. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, New York, NY, USA, 2021. Association for Computing Machinery.

[47] Carrie J. Cai, Emily Reif, Narayan Hegde, Jason Hipp, Been Kim, Daniel Smilkov, Martin Wattenberg, Fernanda Viegas, Greg S. Corrado, Martin C. Stumpe, and Michael Terry. Human-centered tools for coping with imperfect algorithms during medical decision-making. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing*

Systems, CHI '19, page 1–14, New York, NY, USA, 2019. Association for Computing Machinery.

[48] Alex Calderwood, Vivian Qiu, Katy Ilonka Gero, and Lydia B Chilton. How novelists use generative language models: An exploratory user study. In *HAI-GEN+ user2agent@ IUI*, 2020.

[49] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[50] Erin A. Carroll, Celine Latulipe, Richard Fung, and Michael Terry. Creativity factor evaluation: Towards a standardized survey metric for creativity support. In *Proceedings of the Seventh ACM Conference on Creativity and Cognition*, C&C '09, page 127–136, New York, NY, USA, 2009. Association for Computing Machinery.

[51] Elin Carstensdottir, Nathan Partlan, Steven Sutherland, Tyler Duke, Erika Ferris, Robin M. Richter, Maria Valladares, and Magy Seif El-Nasr. Progression maps: Conceptualizing narrative structure for interaction design support. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[52] Marc Cavazza, Fred Charles, and Steven J. Mead. Characters in search of an author: Ai-based virtual storytelling. In *Proceedings of the International Conference on Virtual Storytelling: Using Virtual Reality Technologies for Storytelling*, ICVS '01, page 145–154, Berlin, Heidelberg, 2001. Springer-Verlag.

[53] Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. Cocon: A self-supervised approach for controlled text generation. In *International Conference on Learning Representations*, 2021.

[54] Joel Chan, Steven Dang, and Steven P. Dow. Improving crowd innovation with expert facilitation. In Darren Gergle, Meredith Ringel Morris, Pernille Bjørn, and Joseph A. Konstan, editors, *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, CSCW 2016, San Francisco, CA, USA, February 27 - March 2, 2016*, pages 1221–1233, New York, USA, 2016. ACM.

[55] Julia Chatain, Olivier Bitter, Violaine Fayolle, Robert W. Sumner, and Stéphane Magnenat. A creative game design and programming app. In *Motion, Interaction and Games*, MIG '19, New York, NY, USA, 2019. Association for Computing Machinery.

[56] Siddhartha Chaudhuri, Evangelos Kalogerakis, Stephen Giguere, and Thomas Funkhouser. Attribit: Content creation with semantic attributes. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, page 193–202, New York, NY, USA, 2013. Association for Computing Machinery.

[57] Siddhartha Chaudhuri and Vladlen Koltun. Data-driven suggestions for creativity support in 3d modeling. *ACM Trans. Graph.*, 29(6), December 2010.

[58] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

[59] Erin Cherry and Celine Latulipe. Quantifying the creativity support of digital tools through the creativity support index. *ACM Trans. Comput.-Hum. Interact.*, 21(4), jun 2014.

[60] Mao-Lin Chiu. An organizational view of design communication in design collaboration. *Design Studies*, 23(2):187 – 210, 2002.

[61] Saemi Choi, Shun Matsumura, and Kiyoharu Aizawa. Assist users' interactions in font search with unexpected but useful concepts generated by multimodal learning. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, ICMR '19, page 235–243, New York, NY, USA, 2019. Association for Computing Machinery.

[62] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[63] John Joon Young Chung, Shiqing He, and Eytan Adar. The intersection of users, roles, interactions, and technologies in creativity support tools. In *Designing Interactive Systems Conference 2021*, DIS '21, page 1817–1833, New York, NY, USA, 2021. Association for Computing Machinery.

[64] John Joon Young Chung, Shiqing He, and Eytan Adar. The intersection of users, roles, interactions, and technologies in creativity support tools. In *Conference on Designing Interactive Systems*, pages 1817–1833. ACM, 2021.

[65] John Joon Young Chung, Shiqing He, and Eytan Adar. Artist support networks: Implications for future creativity support tools. In *Designing Interactive Systems Conference*, DIS '22, page 232–246, New York, NY, USA, 2022. Association for Computing Machinery.

[66] John Joon Young Chung, Wooseok Kim, Kang Min Yoo, Hwaran Lee, Eytan Adar, and Minsuk Chang. *TaleBrush: Sketching Stories with Generative Pretrained Language Models*. Association for Computing Machinery, New York, NY, USA, 2022.

[67] Marianela Ciolfi Felice, Sarah Fdili Alaoui, and Wendy E. Mackay. Knotation: Exploring and documenting choreographic processes. In *Proceedings of the 2018 CHI Conference on*

Human Factors in Computing Systems, CHI '18, page 1–12, New York, NY, USA, 2018. Association for Computing Machinery.

[68] A. Clark and A. Dünser. An interactive augmented reality coloring book. In *2012 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 7–10, 2012.

[69] Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. Creative writing with a machine in the loop: Case studies on slogans and stories. In *23rd International Conference on Intelligent User Interfaces*, IUI '18, page 329–340, New York, NY, USA, 2018. Association for Computing Machinery.

[70] Andy Coenen, Luke Davis, Daphne Ippolito, Emily Reif, and Ann Yuan. Wordcraft: a human-ai collaborative editor for story writing. *arXiv preprint arXiv:2107.07430*, 2021.

[71] W.W. Cook and P. Collins. *Plotto: The Master Book of All Plots*. Tin House Books, 2016.

[72] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance, 2022.

[73] Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance, 2022.

[74] Hai Dang, Lukas Mecke, and Daniel Buschek. Ganslider: How users control generative models for images using multiple sliders with and without feedforward information. *arXiv preprint arXiv:2202.00965*, 2022.

[75] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020.

[76] Stephen Davies. *Definitions of art*. Cornell University Press, USA, 1991.

[77] Nicholas Davis, Chih-Pin Hsiao, Yanna Popova, and Brian Magerko. *An Enactive Model of Creativity for Computational Collaboration and Co-creation*, pages 109–133. Springer London, London, 2015.

[78] Nicholas Davis, Chih-PIn Hsiao, Kunwar Yashraj Singh, Lisa Li, Sanat Moningi, and Brian Magerko. Drawing apprentice: An enactive co-creative agent for artistic collaboration. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*, C&C '15, page 185–186, New York, NY, USA, 2015. Association for Computing Machinery.

[79] Nicholas Davis, Chih-PIn Hsiao, Kunwar Yashraj Singh, Lisa Li, and Brian Magerko. Empirically studying participatory sense-making in abstract drawing with a co-creative cognitive agent. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, IUI '16, page 196–207, New York, NY, USA, 2016. Association for Computing Machinery.

[80] Nicholas Davis, Alexander Zook, Brian O'Neill, Brandon Headrick, Mark Riedl, Ashton Grosz, and Michael Nitsche. Creativity support for novice digital filmmaking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 651–660, New York, NY, USA, 2013. Association for Computing Machinery.

[81] Richard De Charms. *Personal causation: The international affective determinants of behavior*. Acad. Press, 1970.

[82] Alexandre Denis, Samuel Cruz-Lara, Nadia Bellalem, and Lotfi Bellalem. Visualization of affect in movie scripts. In *Empatex, 1st International Workshop on Empathic Television Experiences at TVX 2014*, Newcastle, United Kingdom, June 2014.

[83] Ruta Desai, Fraser Anderson, Justin Matejka, Stelian Coros, James McCann, George Fitzmaurice, and Tovi Grossman. Geppetto: Enabling semantic design of expressive robot behaviors. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–14, New York, NY, USA, 2019. Association for Computing Machinery.

[84] Sebastian Deterding, Jonathan Hook, Rebecca Fiebrink, Marco Gillies, Jeremy Gow, Memo Akten, Gillian Smith, Antonios Liapis, and Kate Compton. Mixed-initiative creative interfaces. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, page 628–635, New York, NY, USA, 2017. Association for Computing Machinery.

[85] Eva Deverell. Character arc plot & kurt vonnegut's story shapes, Mar 2020.

[86] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc., 2021.

[87] Florent Di Bartolo. Performative meshes: Musical expression in visuals. In *Proceedings of the 9th International Conference on Digital and Interactive Arts*, ARTECH 2019, New York, NY, USA, 2019. Association for Computing Machinery.

[88] Daniel Dixon, Manoj Prasad, and Tracy Hammond. Icandraw: Using sketch recognition and corrective feedback to assist a user in drawing human faces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, page 897–906, New York, NY, USA, 2010. Association for Computing Machinery.

[89] Chris Donahue, Mina Lee, and Percy Liang. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online, July 2020. Association for Computational Linguistics.

[90] Alexander Dumbadze and Suzanne Hudson. *Contemporary Art: 1989 to the present*. John Wiley & Sons, 2012.

[91] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

[92] Denis Dutton. Authenticity in art. In Jerrold Levinson, editor, *The Oxford Handbook of Aesthetics*, pages 258–274. Oxford University Press, UK, 2003.

[93] Alexandre Duval, Thomas Lamson, Gaël de Léséleuc de Kérouara, and Matthias Gallé. Breaking writer's block: Low-cost fine-tuning of natural language generation models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 278–287, Online, April 2021. Association for Computational Linguistics.

[94] Jane L. E, Ohad Fried, and Maneesh Agrawala. Optimizing portrait lighting at capture-time using a 360 camera as a light probe. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, page 221–232, New York, NY, USA, 2019. Association for Computing Machinery.

[95] Jane L. E, Ohad Fried, Jingwan Lu, Jianming Zhang, Radomír Mech, Jose Echevarria, Pat Hanrahan, and James A. Landay. Adaptive photographic composition guidance. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[96] Claudia M Eckert, Nigel Cross, and Jeffrey H Johnson. Intelligent support for communication in design teams: garment shape specifications in the knitwear industry. *Design Studies*, 21(1):99 – 112, 2000.

[97] Philipp Eichmann and Emanuel Zgraggen. Evaluating subjective accuracy in time series pattern-matching using human-annotated rankings. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, IUI '15, page 28–37, New York, NY, USA, 2015. Association for Computing Machinery.

[98] Elliot W Eisner. What do children learn when they paint? *Art Education*, 31(3):6–11, 1978.

[99] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012.

[100] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. Photosketch: A sketch based image query and compositing system. In *SIGGRAPH 2009: Talks*, SIGGRAPH '09, New York, NY, USA, 2009. Association for Computing Machinery.

[101] Nada Endrissat, Gazi Islam, and Claus Noppeney. Visual organizing: Balancing coordination and creative freedom via mood boards. *Journal of Business Research*, 69(7):2353 – 2362, 2016.

[102] Sandra Erdelez. Information encountering: It's more than just bumping into information. *Bulletin of the American Society for Information Science and Technology*, 25(3):26–29, 1999.

[103] Jerry Alan Fails and Dan R. Olsen. Interactive machine learning. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, IUI '03, page 39–45, New York, NY, USA, 2003. Association for Computing Machinery.

[104] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[105] Angela Fan, Mike Lewis, and Yann Dauphin. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy, July 2019. Association for Computational Linguistics.

[106] Chaoran Fan, Kresimir Matkovic, and Helwig Hauser. Sketch-based fast and accurate querying of time series using parameter-sharing lstm networks. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–1, 2020.

[107] Adam M. Fass, Eric A. Bier, and Eytan Adar. Picturepiper: Using a re-configurable pipeline to find images on the web. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, page 51–62, New York, NY, USA, 2000. Association for Computing Machinery.

[108] Jennifer Fernquist, Tovi Grossman, and George Fitzmaurice. Sketch-sketch revolution: An engaging tutorial system for guided sketching and application learning. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, page 373–382, New York, NY, USA, 2011. Association for Computing Machinery.

[109] Fictional Devices. Plottr, 2021.

[110] G Fischer. Symmetry of ignorance, social creativity, and meta-design. *Knowledge-Based Systems*, 13(7):527 – 537, 2000.

[111] Linda Flower and John R. Hayes. A cognitive process theory of writing. *College Composition and Communication*, 32(4):365–387, 1981.

[112] James Fogarty, Desney Tan, Ashish Kapoor, and Simon Winder. Cueflik: Interactive concept learning in image search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, page 29–38, New York, NY, USA, 2008. Association for Computing Machinery.

[113] Camilo Fosco, Vincent Casser, Amish Kumar Bedi, Peter O'Donovan, Aaron Hertzmann, and Zoya Bylinskii. Predicting visual importance across graphic design types. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 249–260, New York, NY, USA, 2020. Association for Computing Machinery.

[114] Allen Foster and Nigel Ford. Serendipity and information seeking: an empirical study. *Journal of documentation*, 2003.

[115] C. Ailie Fraser, Joy O. Kim, Hijung Valentina Shin, Joel Brandt, and Mira Dontcheva. Temporal segmentation of creative live streams. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–12, New York, NY, USA, 2020. Association for Computing Machinery.

[116] C. Ailie Fraser, Julia M. Markel, N. James Basa, Mira Dontcheva, and Scott Klemmer. Remap: Lowering the barrier to help-seeking with multimodal search. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 979–986, New York, NY, USA, 2020. Association for Computing Machinery.

[117] C. Ailie Fraser, Tricia J. Ngoon, Mira Dontcheva, and Scott Klemmer. Replay: Contextually presenting learning videos across software applications. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA, 2019. Association for Computing Machinery.

[118] Jonas Frich, Michael Mose Biskjaer, Lindsay MacDonald Vermeulen, Christian Remy, and Peter Dalsgaard. Strategies in creative professionals' use of digital tools across domains. In *Proceedings of the 2019 on Creativity and Cognition*, C&C '19, page 210–221, New York, NY, USA, 2019. Association for Computing Machinery.

[119] Jonas Frich, Lindsay MacDonald Vermeulen, Christian Remy, Michael Mose Biskjaer, and Peter Dalsgaard. Mapping the landscape of creativity support tools in hci. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–18, New York, NY, USA, 2019. Association for Computing Machinery.

[120] Jonas Frich, Michael Mose Biskjaer, and Peter Dalsgaard. Twenty years of creativity research in human-computer interaction: Current state and future directions. In *Proceedings of the 2018 Designing Interactive Systems Conference*, DIS '18, page 1235–1257, New York, NY, USA, 2018. Association for Computing Machinery.

[121] Emma Frid, Celso Gomes, and Zeyu Jin. Music creation by example. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[122] Kaori Fujinami, Mami Kosaka, and Bipin Indurkhya. Painting an apple with an apple: A tangible tabletop interface for painting with physical objects. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(4), December 2018.

[123] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion, 2022.

[124] Quentin Galvane, Marc Christie, Chrsitophe Lino, and Rémi Ronfard. Camera-on-rails: Automated computation of constrained camera paths. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, MIG '15, page 151–157, New York, NY, USA, 2015. Association for Computing Machinery.

[125] Dario Gamboni. Visual ambiguity and interpretation. *RES: Anthropology and Aesthetics*, (41):5–15, 2002.

[126] Jérémie Garcia, Theophanis Tsandilas, Carlos Agon, and Wendy E. Mackay. Structured observation with polyphony: A multifaceted tool for studying music composition. In *Proceedings of the 2014 Conference on Designing Interactive Systems*, DIS '14, page 199–208, New York, NY, USA, 2014. Association for Computing Machinery.

[127] Monica J. Garfield. *Creativity Support Systems*, pages 745–758. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[128] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, 2016.

[129] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015.

[130] Katy Ilonka Gero and Lydia B. Chilton. How a stylistic, machine-generated thesaurus impacts a writer's process. In *Proceedings of the 2019 on Creativity and Cognition*, C&C '19, page 597–603, New York, NY, USA, 2019. Association for Computing Machinery.

[131] Katy Ilonka Gero and Lydia B. Chilton. Metaphoria: An algorithmic companion for metaphor creation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery.

[132] Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. Story plot generation based on cbr. *Know.-Based Syst.*, 18(4–5):235–242, August 2005.

[133] Florian Geyer, Jochen Budzinski, and Harald Reiterer. Ideavis: A hybrid workspace and interactive visualization for paper-based collaborative sketching sessions. In *Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design*, NordiCHI '12, page 331–340, New York, NY, USA, 2012. Association for Computing Machinery.

[134] Sanchita Ghose and John Jeffrey Prevost. Autofoley: Artificial synthesis of synchronized sound tracks for silent videos with deep learning. *IEEE Transactions on Multimedia*, 23:1895–1907, 2021.

[135] Karni Gilon, Joel Chan, Felicia Y. Ng, Hila Lifshitz-Assaf, Aniket Kittur, and Dafna Shahaf. Analogy mining for specific design needs. In Regan L. Mandryk, Mark Hancock, Mark Perry, and Anna L. Cox, editors, *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, Montreal, QC, Canada, April 21-26, 2018*, page 121, New York, NY, USA, 2018. ACM.

[136] Barney G. Glaser. Basics of grounded theory analysis: Emergence vs. forcing. 1992.

[137] Barney G. Glaser and Anselm L. Strauss. *The Discovery of Grounded Theory: Strategies for Qualitative Research*. Aldine de Gruyter, New York, NY, 1967.

[138] E.H. Gombrich. *The Story of Art - 16th Edition*. Phaidon Press, 1995.

[139] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[140] Leo A. Goodman. Snowball sampling. *The Annals of Mathematical Statistics*, 32(1):148–170, 1961.

[141] Uttam Grandhi and Ina Yosun Chang. Playgami: Augmented reality origami creativity platform. In *ACM SIGGRAPH 2019 Appy Hour*, SIGGRAPH '19, New York, NY, USA, 2019. Association for Computing Machinery.

[142] Nick Greer, Jaime Teevan, and Shamsi Iqbal. An introduction to technological support for writing. Technical Report MSR-TR-2016-1, April 2016.

[143] Garth Griffin and Robert Jacob. Priming creativity through improvisation on an adaptive musical instrument. In *Proceedings of the 9th ACM Conference on Creativity & Cognition*, C&C '13, page 146–155, New York, NY, USA, 2013. Association for Computing Machinery.

[144] Mark D. Gross and Ellen Yi-Luen Do. Ambiguous intentions: A paper-like interface for creative design. In *ACM Symposium on User Interface Software and Technology*, pages 183–192. ACM, 1996.

[145] Jian Guan, Zhexin Zhang, Zhuoer Feng, Zitao Liu, Wenbiao Ding, Xiaoxi Mao, Changjie Fan, and Minlie Huang. Openmeva: A benchmark for evaluating open-ended story generation metrics, 2021.

[146] Anhong Guo, Anuraag Jain, Shomiron Ghose, Gierad Laput, Chris Harrison, and Jeffrey P. Bigham. Crowd-ai camera sensing in the real world. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(3), September 2018.

[147] Matthew Guzdial, Nicholas Liao, Jonathan Chen, Shao-Yu Chen, Shukan Shah, Vishwa Shah, Joshua Reno, Gillian Smith, and Mark O. Riedl. Friend, collaborator, student, manager: How design of an ai-driven game level editor affects creators. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA, 2019. Association for Computing Machinery.

[148] Matthew Guzdial and Mark Riedl. An interaction framework for studying co-creative ai, 2019.

[149] Joshua Hailpern, Erik Hinterbichler, Caryn Leppert, Damon Cook, and Brian P. Bailey. Team storm: Demonstrating an interaction model for working with multiple ideas during creative group work. In *Proceedings of the 6th ACM SIGCHI Conference on Creativity & Cognition*, C&C '07, page 193–202, New York, NY, USA, 2007. Association for Computing Machinery.

[150] Megan K. Halpern, Jakob Tholander, Max Evjen, Stuart Davis, Andrew Ehrlich, Kyle Schustak, Eric P.S. Baumer, and Geri Gay. Moboogie: Creative expression through whole body musical interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, page 557–560, New York, NY, USA, 2011. Association for Computing Machinery.

[151] Yuma Hamasaki, Seiichi Serikawa, and Yuhki Kitazono. Storage and playback device for creative dance with kinect. In *Proceedings of the 7th ACIS International Conference on Applied Computing and Information Technology*, ACIT 2019, New York, NY, USA, 2019. Association for Computing Machinery.

[152] Steve Haroz, Robert Kosara, and Steven L. Franconeri. The connected scatterplot for presenting paired time series. *IEEE Transactions on Visualization and Computer Graphics*, 22(9):2174–2186, 2016.

[153] Brent Harrison, Christopher Purdy, and Mark Riedl. Toward automated story generation with markov chain monte carlo methods and deep neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 13(2):191–197, Jun. 2021.

[154] Devamanyu Hazarika, Mahdi Namazifar, and Dilek Hakkani-Tür. Zero-shot controlled generation with encoder-decoder transformers, 2021.

[155] Shiqing He and Eytan Adar. Plotting with thread: Fabricating delicate punch needle embroidery with x-y plotters. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference*, DIS '20, page 1047–1057, New York, NY, USA, 2020. Association for Computing Machinery.

[156] Jeffrey Heer. Agency plus automation: Designing artificial intelligence into interactive systems. *Proceedings of the National Academy of Sciences*, 116(6):1844–1850, 2019.

[157] Jeffrey Heer and Ben Shneiderman. Interactive dynamics for visual analysis. *Commun. ACM*, 55(4):45–54, April 2012.

[158] Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill™: A bayesian skill rating system. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, page 569–576, Cambridge, MA, USA, 2006. MIT Press.

[159] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. 2022.

[160] Sean Hickey. Bricoleur: A tool for tinkering with programmable video and audio. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI EA '19, page 1–6, New York, NY, USA, 2019. Association for Computing Machinery.

[161] Will E. Hipson and Saif M. Mohammad. Emotion dynamics in movie dialogues, 2021.

[162] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[163] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022.

[164] Rania Hodhod and Brian Magerko. Closing the cognitive gap between humans and interactive narrative agents using shared mental models. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, IUI '16, page 135–146, New York, NY, USA, 2016. Association for Computing Machinery.

[165] Megan Hofmann, Jennifer Mankoff, and Scott E. Hudson. Knitgist: A programming synthesis toolkit for generating functional machine-knitting textures. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 1234–1247, New York, NY, USA, 2020. Association for Computing Machinery.

[166] Christian Holz and Steven Feiner. Relaxed selection techniques for querying time-series graphs. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, page 213–222, New York, NY, USA, 2009. Association for Computing Machinery.

[167] Eric Hoyt, Kevin Ponto, and Carrie Roy. Visualizing and analyzing the hollywood screenplay with scripthreads. *Digit. Humanit. Q.*, 8(4), 2014.

[168] Chi-yang Hsu, Yun-Wei Chu, Ting-Hao Huang, and Lun-Wei Ku. Plot and rework: Modeling storylines for visual storytelling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4443–4453, Online, August 2021. Association for Computational Linguistics.

[169] Ting-Yao Hsu, Yen-Chia Hsu, and Ting-Hao (Kenneth) Huang. On how users edit computer-generated visual stories. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI EA '19, page 1–6, New York, NY, USA, 2019. Association for Computing Machinery.

[170] Ting-Yao Hsu, Chieh-Yang Huang, Yen-Chia Hsu, and Ting-Hao Huang. Visual story post-editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6581–6586, Florence, Italy, July 2019. Association for Computational Linguistics.

[171] Stacy Hsueh, Sarah Fdili Alaoui, and Wendy E. Mackay. Understanding kinaesthetic creativity in dance. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery.

[172] Cheng-Zhi Anna Huang, David Duvenaud, and Krzysztof Z. Gajos. Chordripple: Recommending chords to help novice composers go beyond the ordinary. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, IUI '16, page 241–250, New York, NY, USA, 2016. Association for Computing Machinery.

[173] Chieh-Yang Huang, Shih-Hong Huang, and Ting-Hao Kenneth Huang. Heteroglossia: In-situ story ideation with the crowd. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[174] Chieh-Yang Huang and Ting-Hao Huang. Semantic frame forecast. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2702–2713, Online, June 2021. Association for Computational Linguistics.

[175] Forrest Huang and John F. Canny. Sketchforme: Composing sketched scenes from text descriptions for interactive applications. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, page 209–220, New York, NY, USA, 2019. Association for Computing Machinery.

[176] Forrest Huang, John F. Canny, and Jeffrey Nichols. *Swire: Sketch-Based User Interface Retrieval*, page 1–10. Association for Computing Machinery, New York, NY, USA, 2019.

[177] Forrest Huang, Eldon Schoop, David Ha, and John Canny. Scones: Towards conversational authoring of sketches. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, IUI '20, page 313–323, New York, NY, USA, 2020. Association for Computing Machinery.

[178] Rongjie Huang, Feiyang Chen, Yi Ren, Jinglin Liu, Chenye Cui, and Zhou Zhao. *Multi-Singer: Fast Multi-Singer Singing Voice Vocoder With A Large-Scale Corpus*, page 3945–3954. Association for Computing Machinery, New York, NY, USA, 2021.

[179] Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1233–1239, San Diego, California, June 2016. Association for Computational Linguistics.

[180] Yichen Huang, Yizhe Zhang, Oussama Elachqar, and Yu Cheng. INSET: Sentence infilling with INter-SEntential transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2502–2515, Online, July 2020. Association for Computational Linguistics.

[181] Jordan S. Huffaker, Jonathan K. Kummerfeld, Walter S. Lasecki, and Mark S. Ackerman. *Crowdsourced Detection of Emotionally Manipulative Language*, page 1–14. Association for Computing Machinery, New York, NY, USA, 2020.

[182] Julie S. Hui, Darren Gergle, and Elizabeth M. Gerber. *IntroAssist: A Tool to Support Writing Introductory Help Requests*, page 1–13. Association for Computing Machinery, New York, NY, USA, 2018.

[183] Emmanuel Iarussi, Adrien Bousseau, and Theophanis Tsandilas. The drawing assistant: Automated drawing guidance and feedback from photographs. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, page 183–192, New York, NY, USA, 2013. Association for Computing Machinery.

[184] Junko Ichino and Hayato Nao. Playing the body: Making music through various body movements. In *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, AM'18, New York, NY, USA, 2018. Association for Computing Machinery.

[185] Daphne Ippolito, David Grangier, Chris Callison-Burch, and Douglas Eck. Unsupervised hierarchical story infilling. In *Proceedings of the First Workshop on Narrative Understanding*, pages 37–43, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[186] Jennifer Jacobs, Joel Brandt, Radomír Mech, and Mitchel Resnick. Extending manual drawing practices with artist-centric programming tools. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–13, New York, NY, USA, 2018. Association for Computing Machinery.

[187] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields, 2021.

[188] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. December 2021.

[189] Yunwoo Jeong, Han-Jong Kim, Gyeongwon Yun, and Tek-Jin Nam. Wika: A projected augmented reality workbench for interactive kinetic art. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 999–1009, New York, NY, USA, 2020. Association for Computing Machinery.

[190] V. John-Steiner. *Creative Collaboration*. Oxford University Press, 2000.

[191] Annamma Joy and Jr. John F. Sherry. Disentangling the paradoxical alliances between art market and art world. *Consumption Markets & Culture*, 6(3):155–181, 2003.

[192] Stefan Jänicke, Greta Franzini, Muhammad Faisal Cheema, and Gerik Scheuermann. On Close and Distant Reading in Digital Humanities: A Survey and Future Challenges. In R. Borgo, F. Ganovelli, and I. Viola, editors, *Eurographics Conference on Visualization (EuroVis) - STARs*. The Eurographics Association, 2015.

[193] Peter H. Kahn, Takayuki Kanda, Hiroshi Ishiguro, Brian T. Gill, Solace Shen, Jolina H. Ruckert, and Heather E. Gary. Human creativity can be facilitated through interacting with a social robot. In *The Eleventh ACM/IEEE International Conference on Human Robot Interaction*, HRI '16, page 173–180, USA, 2016. IEEE Press.

[194] Hiroki Kaimoto, Junichi Yamaoka, Satoshi Nakamaru, Yoshihiro Kawahara, and Yasuaki Kakehi. Expandfab: Fabricating objects expanding and changing shape with heat. In *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '20, page 153–164, New York, NY, USA, 2020. Association for Computing Machinery.

[195] Jermain Kaminski, Michael Schober, Raymond Albaladejo, Oleksandr Zastupailo, and César Hidalgo. Moviegalaxies-social networks in movies. 2018.

[196] Pegah Karimi, Nicholas Davis, Mary Lou Maher, Kazjon Grace, and Lina Lee. Relating cognitive models of design creativity to the similarity of sketches generated by an ai partner. In *Proceedings of the 2019 on Creativity and Cognition*, C&C '19, page 259–270, New York, NY, USA, 2019. Association for Computing Machinery.

[197] Pegah Karimi, Jeba Rezwana, Safat Siddiqui, Mary Lou Maher, and Nasrin Dehbozorgi. Creative sketching partner: An analysis of human-ai co-creativity. In *Proceedings of the 25th International Conference on Intelligent User Interfaces*, IUI '20, page 221–230, New York, NY, USA, 2020. Association for Computing Machinery.

[198] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019.

[199] Joseph Kasof, Chuansheng Chen, Amy Himsel, and Ellen Greenberger. Values and creativity. *Creativity Research Journal*, 19(2-3):105–122, 2007.

[200] Jun Kato, Tomoyasu Nakano, and Masataka Goto. Textalive: Integrated design environment for kinetic typography. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, page 3403–3412, New York, NY, USA, 2015. Association for Computing Machinery.

[201] Natsumi Kato, Hiroyuki Osone, Daitetsu Sato, Naoya Muramatsu, and Yoichi Ochiai. Deepwear: A case study of collaborative design between human and artificial intelligence. In *Proceedings of the Twelfth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '18, page 529–536, New York, NY, USA, 2018. Association for Computing Machinery.

[202] D. Kaufer, C. Geisler, Pantelis Vlachos, and S. Ishizaki. Mining textual knowledge for writing education and research: The docuscope project. *Studies in Writing*, pages 115–129, 01 2006.

[203] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models, 2022.

[204] Rubaiat Habib Kazi, Kien Chuan Chua, Shengdong Zhao, Richard Davis, and Kok-Lim Low. Sandcanvas: A multi-touch art medium inspired by sand animation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, page 1283–1292, New York, NY, USA, 2011. Association for Computing Machinery.

[205] Eamonn Keogh and Padhraic Smyth. A probabilistic approach to fast pattern matching in time series databases. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, KDD'97, page 24–30. AAAI Press, 1997.

[206] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation, 2019.

[207] Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. A distributional approach to controlled text generation. In *International Conference on Learning Representations*, 2021.

[208] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2668–2677, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[209] Joy Kim, Maneesh Agrawala, and Michael S. Bernstein. Mosaic: Designing online creative communities for sharing works-in-progress. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, page 246–258, New York, NY, USA, 2017. Association for Computing Machinery.

[210] Joy Kim, Justin Cheng, and Michael S. Bernstein. Ensemble: Exploring complementary strengths of leaders and crowds in creative collaboration. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '14, page 745–755, New York, NY, USA, 2014. Association for Computing Machinery.

[211] Joy Kim, Mira Dontcheva, Wilmot Li, Michael S. Bernstein, and Daniela Steinsapir. Motif: Supporting novice creativity through expert patterns. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, page 1211–1220, New York, NY, USA, 2015. Association for Computing Machinery.

[212] Joy Kim, Sarah Sterman, Allegra Argent Beal Cohen, and Michael S. Bernstein. Mechanical novel: Crowdsourcing complex work through reflection and revision. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, page 233–245, New York, NY, USA, 2017. Association for Computing Machinery.

[213] Nam Wook Kim, Benjamin Bach, Hyejin Im, Sasha Schriber, Markus Gross, and Hanspeter Pfister. Visualizing nonlinear narratives with story curves. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):595–604, 2018.

[214] Tae Soo Kim, DaEun Choi, Yoonseo Choi, and Juho Kim. Stylette: Styling the web with natural language. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery.

[215] Taewook Kim, Jung Soo Lee, Zhenhui Peng, and Xiaojuan Ma. Love in lyrics: An exploration of supporting textual manifestation of affection in social messaging. *Proc. ACM Hum.-Comput. Interact.*, 3(CSCW), nov 2019.

[216] Vladimir G. Kim, Wilmot Li, Niloy J. Mitra, Stephen DiVerdi, and Thomas Funkhouser. Exploring collections of 3d models using fuzzy correspondences. *ACM Trans. Graph.*, 31(4), July 2012.

[217] Yea-Seul Kim, Mira Dontcheva, Eytan Adar, and Jessica Hullman. Vocal shortcuts for creative experts. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–14, New York, NY, USA, 2019. Association for Computing Machinery.

[218] Janin Koch, Andrés Lucero, Lena Hegemann, and Antti Oulasvirta. May ai? design ideation with cooperative contextual bandits. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery.

[219] Janin Koch, Nicolas Taffin, Michel Beaudouin-Lafon, Markku Laine, Andrés Lucero, and Wendy E. Mackay. Imagesense: An intelligent collaborative ideation tool to support diverse human-computer partnerships. *Proc. ACM Hum.-Comput. Interact.*, 4(CSCW1), May 2020.

[220] Janin Koch, Nicolas Taffin, Andrés Lucero, and Wendy E. Mackay. *SemanticCollage: Enriching Digital Mood Board Design with Semantic Labels*, page 407–418. Association for Computing Machinery, New York, NY, USA, 2020.

[221] Xiangzhe Kong, Jialiang Huang, Ziquan Tung, Jian Guan, and Minlie Huang. Stylized story generation with style-guided planning, 2021.

[222] Yuki Koyama, Daisuke Sakamoto, and Takeo Igarashi. Crowd-powered parameter analysis for visual design exploration. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, page 65–74, New York, NY, USA, 2014. Association for Computing Machinery.

[223] Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. Gedi: Generative discriminator guided sequence generation, 2020.

[224] Max Kreminski, Devi Acharya, Nick Junius, Elisabeth Oliver, Kate Compton, Melanie Dickinson, Cyril Focht, Stacey Mason, Stella Mazeika, and Noah Wardrip-Fruin. Cozy mystery construction kit: Prototyping toward an ai-assisted collaborative storytelling mystery game. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, FDG '19, New York, NY, USA, 2019. Association for Computing Machinery.

[225] Max Kreminski, Melanie Dickinson, Michael Mateas, and Noah Wardrip-Fruin. Why are we like this?: The ai architecture of a co-creative storytelling game. In *International Conference on the Foundations of Digital Games*, FDG '20, New York, NY, USA, 2020. Association for Computing Machinery.

[226] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[227] Gihyun Kwon and Jong Chul Ye. Clipstyler: Image style transfer with a single text condition, 2021.

[228] Pierre-Yves Laffont, Zhile Ren, Xiaofeng Tao, Chao Qian, and James Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Trans. Graph.*, 33(4), July 2014.

[229] J.A. Landay and B.A. Myers. Sketching interfaces: toward more human interface design. *Computer*, 34(3):56–64, 2001.

[230] Maria Larsson, Hironori Yoshida, Nobuyuki Umetani, and Takeo Igarashi. Tsugite: Interactive design and fabrication of wood joints. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 317–327, New York, NY, USA, 2020. Association for Computing Machinery.

[231] Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. *Automated Grammatical Error Detection for Language Learners*. Morgan and Claypool Publishers, 2010.

[232] Michael Lebowitz. Creating a story-telling universe. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'83, page 63–65, San Francisco, CA, USA, 1983. Morgan Kaufmann Publishers Inc.

[233] Brian Lee, Savil Srivastava, Ranjitha Kumar, Ronen Brafman, and Scott R. Klemmer. *Designing with Interactive Example Galleries*, page 2257–2266. Association for Computing Machinery, New York, NY, USA, 2010.

[234] Charlotte P. Lee. Boundary negotiating artifacts: Unbinding the routine of boundary objects and embracing chaos in collaborative work. *Comput. Supported Coop. Work*, 16(3):307–339, June 2007.

[235] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021.

[236] Jiahao Li, Meilin Cui, Jeeeun Kim, and Xiang 'Anthony' Chen. Romeo: A design tool for embedding transformable parts in 3d models to robotically augment default functionalities. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 897–911, New York, NY, USA, 2020. Association for Computing Machinery.

[237] Jingyi Li, Joel Brandt, Radomír Mech, Maneesh Agrawala, and Jennifer Jacobs. Supporting visual artists in programming through direct inspection and control of program execution. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–12, New York, NY, USA, 2020. Association for Computing Machinery.

[238] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation, 2021.

[239] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 385–395, Red Hook, NY, USA, 2017. Curran Associates Inc.

[240] Jun Hao Liew, Hanshu Yan, Daquan Zhou, and Jiashi Feng. Magicmix: Semantic mixing with diffusion models. *arXiv preprint arXiv:2210.16056*, 2022.

[241] Alex Limpaecher, Nicolas Feltman, Adrien Treuille, and Michael Cohen. Real-time drawing assistance through crowdsourcing. *ACM Trans. Graph.*, 32(4), July 2013.

[242] Shih-Ting Lin, Nathanael Chambers, and Greg Durrett. Conditional generation of temporally-ordered event sequences, 2021.

[243] Yuyu Lin, Jiahao Guo, Yang Chen, Cheng Yao, and Fangtian Ying. It is your turn: Collaborative ideation with a co-creative robot through sketch. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–14, New York, NY, USA, 2020. Association for Computing Machinery.

[244] Zhiyu Lin and Mark Riedl. Plug-and-blend: A framework for controllable story generation with blended control codes, 2021.

[245] Lasse Lingens, Robert W. Sumner, and Stéphane Magnenat. Towards automatic drawing animation using physics-based evolution. In *Proceedings of the 2020 ACM Interaction Design and Children Conference: Extended Abstracts*, IDC '20, page 314–319, New York, NY, USA, 2020. Association for Computing Machinery.

[246] Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online, August 2021. Association for Computational Linguistics.

[247] Jingyuan Liu, Hongbo Fu, and Chiew-Lan Tai. Posetween: Pose-driven tween animation. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 791–804, New York, NY, USA, 2020. Association for Computing Machinery.

[248] Lucas Liu, Duri Long, Swar Gujrania, and Brian Magerko. Learning movement through human-computer co-creative improvisation. In *Proceedings of the 6th International Conference on Movement and Computing*, MOCO '19, New York, NY, USA, 2019. Association for Computing Machinery.

[249] Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B. Tenenbaum. Compositional visual generation with composable diffusion models, 2022.

[250] Shixia Liu, Yingcai Wu, Enxun Wei, Mengchen Liu, and Yang Liu. Storyflow: Tracking the evolution of stories. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2436–2445, 2013.

[251] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too, 2021.

[252] Duri Long, Mikhail Jacob, and Brian Magerko. Designing co-creative ai for public spaces. In *Proceedings of the 2019 on Creativity and Cognition*, C&C '19, page 271–284, New York, NY, USA, 2019. Association for Computing Machinery.

[253] Duri Long, Lucas Liu, Swar Gujrania, Cassandra Naomi, and Brian Magerko. Visualizing improvisation in luminai, an ai partner for co-creative dance. In *Proceedings of the 7th International Conference on Movement and Computing*, MOCO '20, New York, NY, USA, 2020. Association for Computing Machinery.

[254] Daniel Lopes, João Correia, and Penousal Machado. Adea - evolving glyphs for aiding creativity in typeface design. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, GECCO '20, page 97–98, New York, NY, USA, 2020. Association for Computing Machinery.

[255] Sandy Louchart and Ruth Aylett. Building synthetic actors for interactive dramas. In *AAAI Fall Symposium: Intelligent Narrative Technologies*, pages 63–70, 2007.

[256] Ryan Louie, Andy Coenen, Cheng Zhi Huang, Michael Terry, and Carrie J. Cai. Novice-ai music co-creation via ai-steering tools for deep generative models. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[257] Brian Lubars and Chenhao Tan. Ask not what ai can do, but what ai should do: Towards a framework of task delegability. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 57–67. Curran Associates, Inc., 2019.

[258] A. Lucero, D. Aliakseyeu, and J. Martens. Augmenting mood boards: Flexible and intuitive interaction in the context of the design studio. In *Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07)*, pages 147–154, 2007.

[259] Andrés Lucero. Framing, aligning, paradoxing, abstracting, and directing: How design mood boards work. In *Proceedings of the Designing Interactive Systems Conference*, DIS '12, page 438–447, New York, NY, USA, 2012. Association for Computing Machinery.

[260] Andrés Lucero and Kirsikka Vaajakallio. Co-designing mood boards: Creating dialogue with people. In *Proceedings of the Third IASTED International Conference on Human Computer Interaction*, HCI '08, page 254–260, USA, 2008. ACTA Press.

[261] Kurt Luther and Amy Bruckman. Leadership in online creative collaboration. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, page 343–352, New York, NY, USA, 2008. Association for Computing Machinery.

[262] Kurt Luther, Kelly Caine, Kevin Ziegler, and Amy Bruckman. Why it works (when it works): Success factors in online creative collaboration. In *Proceedings of the 16th ACM International Conference on Supporting Group Work*, GROUP '10, page 1–10, New York, NY, USA, 2010. Association for Computing Machinery.

[263] Kurt Luther, Casey Fiesler, and Amy Bruckman. Redistributing leadership in online creative collaboration. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, CSCW '13, page 1007–1022, New York, NY, USA, 2013. Association for Computing Machinery.

[264] Sam Ross Lydia B. Chilton, Ecenaz Jen Ozmen. Visifit: Ai tools to iteratively improve visual blends, 2019.

[265] N. Mahyar and M. Tory. Supporting communication and coordination in collaborative sensemaking. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1633–1642, 2014.

[266] Neil Maiden, Konstantinos Zachos, Amanda Brown, George Brock, Lars Nyre, Aleksander Nygård Tonheim, Dimitris Apsotolou, and Jeremy Evans. *Making the News: Digital Creativity Support for Journalists*, page 1–11. Association for Computing Machinery, New York, NY, USA, 2018.

[267] Lena Mamykina, Linda Candy, and Ernest Edmonds. Collaborative creativity. *Commun. ACM*, 45(10):96–99, October 2002.

[268] Miro Mannino and Azza Abouzied. *Expressive Time Series Querying with Hand-Drawn Scale-Free Sketches*, page 1–13. Association for Computing Machinery, New York, NY, USA, 2018.

[269] Miro Mannino and Azza Abouzied. Is this real? generating synthetic data that looks real. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, page 549–561, New York, NY, USA, 2019. Association for Computing Machinery.

[270] Stacy C. Marsella, W. Lewis Johnson, and Catherine LaBore. Interactive pedagogical drama. In *Proceedings of the Fourth International Conference on Autonomous Agents*, AGENTS '00, page 301–308, New York, NY, USA, 2000. Association for Computing Machinery.

[271] Lara Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark Riedl. Event representations for automated story generation with deep neural nets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[272] C. Martindale. *Clockwork Muse*. Basic Books, 1990.

[273] Dimitri Masson, Alexandre Demeure, and Gaelle Calvary. Magellan, an evolutionary system to foster user interface design creativity. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '10, page 87–92, New York, NY, USA, 2010. Association for Computing Machinery.

[274] MasterClass. Learn about narrative arcs: Definition, examples, and how to create a narrative arc in your writing - 2021, Nov 2020.

[275] Justin Matejka, Michael Glueck, Erin Bradner, Ali Hashemi, Tovi Grossman, and George Fitzmaurice. Dream lens: Exploration and visualization of large-scale generative design datasets. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–12, New York, NY, USA, 2018. Association for Computing Machinery.

[276] Jon McCormack, Toby Gifford, and Patrick Hutchings. Autonomy, authenticity, authorship and intention in computer generated art. In *Computational Intelligence in Music, Sound, Art and Design: 8th International Conference, EvoMUSART 2019, Held as Part of EvoStar 2019, Leipzig, Germany, April 24–26, 2019, Proceedings*, page 35–50, Berlin, Heidelberg, 2019. Springer-Verlag.

[277] Jon McCormack, Toby Gifford, Patrick Hutchings, Maria Teresa Llano Rodriguez, Matthew Yee-King, and Mark d'Inverno. In a silent way: Communication between ai and improvising musicians beyond sound. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–11, New York, NY, USA, 2019. Association for Computing Machinery.

[278] Deana Mcdonagh and Ian Storer. Mood boards as a design catalyst and resource: Researching an under-researched area. *The Design Journal*, 7(3):16–31, 2004.

[279] James R. Meehan. Tale-spin, an interactive program that writes stories. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'77, page 91–98, San Francisco, CA, USA, 1977. Morgan Kaufmann Publishers Inc.

[280] Microsoft Research. Trueskill, 2017.

[281] K. L. Bhanu Moorthy, Moneish Kumar, Ramanathan Subramanian, and Vineet Gandhi. Gazed– gaze-guided cinematic editing of wide-angle monocular video recordings. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–11, New York, NY, USA, 2020. Association for Computing Machinery.

[282] Franco Moretti. *Graphs, maps, trees: abstract models for a literary history*. Verso, 2005.

[283] Yusuke Mori, Hiroaki Yamane, Yusuke Mukuta, and Tatsuya Harada. Finding and generating a missing part for story completion. In *Proceedings of the The 4th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 156–166, Online, December 2020. International Committee on Computational Linguistics.

[284] M. R. Morris, A. Paepcke, and T. Winograd. Teamsearch: comparing techniques for co-present collaborative search of digital media. In *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP '06)*, pages 8 pp.–, 2006.

[285] Meredith Ringel Morris. A survey of collaborative web search practices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, page 1657–1660, New York, NY, USA, 2008. Association for Computing Machinery.

[286] Meredith Ringel Morris and Eric Horvitz. Searchtogether: An interface for collaborative web search. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, page 3–12, New York, NY, USA, 2007. Association for Computing Machinery.

[287] Meredith Ringel Morris, Jarrod Lombardo, and Daniel Wigdor. Wesearch: Supporting collaborative search and sensemaking on a tabletop display. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, CSCW '10, page 401–410, New York, NY, USA, 2010. Association for Computing Machinery.

[288] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California, June 2016. Association for Computational Linguistics.

[289] Michael Muller and S. Kogan. Grounded theory method in hci and cscw. 01 2010.

[290] T. Munzner. *Visualization Analysis and Design*. AK Peters Visualization Series. CRC Press, 2015.

[291] Brad A. Myers, Ashley Lai, Tam Minh Le, YoungSeok Yoon, Andrew Faulring, and Joel Brandt. Selective undo support for painting applications. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, page 4227–4236, New York, NY, USA, 2015. Association for Computing Machinery.

[292] Kumiyo Nakakoji. Meanings of tools, support, and uses for creative design processes. *International design research symposium '06*, pages 156–165, 12 2006.

[293] Eric T. Nalisnick and Henry S. Baird. Character-to-character sentiment analysis in shakespeare's plays. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 479–483, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[294] Timothy Neate, Abi Roper, Stephanie Wilson, Jane Marshall, and Madeline Cruice. Creatable content and tangible interaction in aphasia. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–14, New York, NY, USA, 2020. Association for Computing Machinery.

[295] Michael Nebeling, Alexandra To, Anhong Guo, Adrian A. de Freitas, Jaime Teevan, Steven P. Dow, and Jeffrey P. Bigham. *WearWrite: Crowd-Assisted Writing from Smartwatches*, page 3834–3846. Association for Computing Machinery, New York, NY, USA, 2016.

[296] George E Newman and Paul Bloom. Art and authenticity: The importance of originals in judgments of value. *Journal of Experimental Psychology: General*, 141(3):558, 2012.

[297] Tricia J. Ngoon, C. Ailie Fraser, Ariel S. Weingarten, Mira Dontcheva, and Scott Klemmer. *Interactive Guidance Techniques for Improving Creative Feedback*, page 1–11. Association for Computing Machinery, New York, NY, USA, 2018.

[298] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[299] Donald A. Norman. *The design of everyday things*. Basic Books, [New York], 2002.

[300] Peter O'Donovan, Jundefinednis Lundefinedbeks, Aseem Agarwala, and Aaron Hertzmann. Exploratory font selection using crowdsourced attributes. *ACM Trans. Graph.*, 33(4), July 2014.

[301] Changhoon Oh, Jungwoo Song, Jinhan Choi, Seonghyeon Kim, Sungwoo Lee, and Bongwon Suh. I lead, you help but only with enough details: Understanding user experience of co-creation with artificial intelligence. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–13, New York, NY, USA, 2018. Association for Computing Machinery.

[302] Santiago Ortiz. Lostalgic, Oct 2012.

[303] Jiefu Ou, Nathaniel Weir, Anton Belyy, Felix Yu, and Benjamin Van Durme. InFillmore: Frame-guided language generation with bidirectional context. In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 129–142, Online, August 2021. Association for Computational Linguistics.

[304] Antti Oulasvirta, Anna Feit, Perttu Lähteenlahti, and Andreas Karrenbauer. Computational support for functionality selection in interaction design. *ACM Trans. Comput.-Hum. Interact.*, 24(5), October 2017.

[305] Maks Ovsjanikov, Wilmot Li, Leonidas Guibas, and Niloy J. Mitra. Exploration of continuous variability in collections of 3d shapes. *ACM Trans. Graph.*, 30(4), July 2011.

[306] Google PAIR. People + ai guidebook, May 2019.

[307] D. Y. Park and K. H. Lee. Arbitrary style transfer with style-attentional networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5873–5881, 2019.

[308] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[309] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[310] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei A. Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. In *Advances in Neural Information Processing Systems*, 2020.

[311] Nathan Partlan, Elin Carstensdottir, Erica Kleinman, Sam Snodgrass, Casper Harteveld, Gillian Smith, Camillia Matuk, Steven C. Sutherland, and Magy Seif El-Nasr. Evaluation of an automatically-constructed graph-based representation for interactive narrative. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, FDG '19, New York, NY, USA, 2019. Association for Computing Machinery.

[312] Bec Paton and Kees Dorst. Briefing and reframing: A situated practice. *Design Studies*, 32(6):573 – 587, 2011. Interpreting Design Thinking.

[313] Sharoda A. Paul and Meredith Ringel Morris. Cosense: Enhancing sensemaking for collaborative web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, page 1771–1780, New York, NY, USA, 2009. Association for Computing Machinery.

[314] J. Scott Penberthy and Daniel S. Weld. Ucpop: A sound, complete, partial order planner for adl. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, KR'92, page 103–114, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

[315] Mengqi Peng, Li-yi Wei, Rubaiat Habib Kazi, and Vladimir G. Kim. Autocomplete animated sculpting. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 760–777, New York, NY, USA, 2020. Association for Computing Machinery.

[316] Zhenhui Peng, Qingyu Guo, Ka Wing Tsang, and Xiaojuan Ma. Exploring the effects of technological writing assistance for support providers in online mental health community. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–15, New York, NY, USA, 2020. Association for Computing Machinery.

[317] Allison Perrone and Justin Edwards. Chatbots as unwitting actors. In *Proceedings of the 1st International Conference on Conversational User Interfaces*, CUI '19, New York, NY, USA, 2019. Association for Computing Machinery.

[318] James L. Peterson. Computer programs for detecting and correcting spelling errors. *Commun. ACM*, 23(12):676–687, December 1980.

[319] Florian Pinel and Lav R. Varshney. Computational creativity for culinary recipes. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '14, page 439–442, New York, NY, USA, 2014. Association for Computing Machinery.

[320] Azzurra Pini, Jer Hayes, Connor Upton, and Medb Corcoran. Ai inspired recipes: Designing computationally creative food combos. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI EA '19, page 1–6, New York, NY, USA, 2019. Association for Computing Machinery.

[321] Cecil Piya, Vinayak , Senthil Chandrasegaran, Niklas Elmqvist, and Karthik Ramani. Co-3deator: A team-first collaborative 3d design ideation tool. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, page 6581–6592, New York, NY, USA, 2017. Association for Computing Machinery.

[322] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv*, 2022.

[323] Ernst Pöppel. *Time Perception*, pages 713–729. Springer Berlin Heidelberg, Berlin, Heidelberg, 1978.

[324] Rafael PÉrez Ý PÉrez and Mike Sharples. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence*, 13(2):119–139, 2001.

[325] Lu Qiang, Chai Bingjie, and Zhang Haibo. Storytelling by the storycake visualization. *Vis. Comput.*, 33(10):1241–1252, October 2017.

[326] Brian Quanz, Wei Sun, Ajay Deshpande, Dhruv Shah, and Jae eun Park. Machine learning based co-creative design framework, 2020.

[327] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[328] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[329] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[330] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

[331] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Mark Chen, Rewon Child, Vedant Misra, Pamela Mishkin, Gretchen Krueger, Sandhini Agarwal, and Ilya Sutskever. Dall·e: Creating images from text, 2021. Accessed: January, 2021.

[332] Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online, November 2020. Association for Computational Linguistics.

[333] Andrew J. Reagan, Lewis Mitchell, Dilan Kiley, Christopher M. Danforth, and Peter Sheridan Dodds. The emotional arcs of stories are dominated by six basic shapes. *EPJ Data Science*, 5(1):31, Nov 2016.

[334] Reedsy. What is a narrative arc? • a guide to storytelling structure, Jul 2017.

[335] Christian Remy, Lindsay MacDonald Vermeulen, Jonas Frich, Michael Mose Biskjaer, and Peter Dalsgaard. Evaluating creativity support tools in hci research. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference*, DIS '20, page 457–476, New York, NY, USA, 2020. Association for Computing Machinery.

[336] Mitchel Resnick, Brad Myers, Kumiyo Nakakoji, Ben Shneiderman, Randy Pausch, Ted Selker, and Mike Eisenberg. Design principles for tools to support creative thinking. Technical report, 2005.

[337] Laria Reynolds and Kyle McDonell. *Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm*. Association for Computing Machinery, New York, NY, USA, 2021.

[338] Mark O Riedl. Vignette-based story planning: Creativity through exploration and retrieval. In *Proceedings of the 5th International Joint Workshop on Computational Creativity*, pages 41–50, 2008.

[339] Mark O. Riedl and R. Michael Young. Narrative planning: Balancing plot and character. *J. Artif. Int. Res.*, 39(1):217–268, September 2010.

[340] Mark Owen Riedl and Vadim Bulitko. Interactive narrative: An intelligent systems approach. *AI Magazine*, 34(1):67, Dec. 2012.

[341] Daniel Ritchie, Ankita Arvind Kejriwal, and Scott R. Klemmer. D.tour: Style-based exploration of design example galleries. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, page 165–174, New York, NY, USA, 2011. Association for Computing Machinery.

[342] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[343] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[344] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. ”grabcut”: Interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH ’04, page 309–314, New York, NY, USA, 2004. Association for Computing Machinery.

[345] Quentin Roy, Futian Zhang, and Daniel Vogel. Automation accuracy is good, but high controllability may be better. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI ’19, page 1–8, New York, NY, USA, 2019. Association for Computing Machinery.

[346] Kathy Ryall, Neal Lesh, Tom Lanning, Darren Leigh, Hiroaki Miyashita, and Shigeru Makino. *QueryLines: Approximate Query for Visual Browsing*, page 1765–1768. Association for Computing Machinery, New York, NY, USA, 2005.

[347] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH ’22, New York, NY, USA, 2022. Association for Computing Machinery.

[348] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022.

[349] Ugo Braga Sangiorgi, François Beuvens, and Jean Vanderdonckt. User interface design by collaborative sketching. In *Proceedings of the Designing Interactive Systems Conference*, DIS ’12, page 378–387, New York, NY, USA, 2012. Association for Computing Machinery.

[350] R.K. Sawyer. *Explaining Creativity: The Science of Human Innovation*. Oxford University Press, USA, 2012.

[351] Thomas Schmidt. Distant reading sentiments and emotions in historic german plays. In *Abstract Booklet, DH_Budapest_2019*, pages 57–60. Budapest, Hungary, September 2019.

[352] Eric Hal Schwartz. Resemble ai can now clone your voice to speak new languages, Oct 2020.

[353] Sarah Schwettmann, Evan Hernandez, David Bau, Samuel Klein, Jacob Andreas, and Antonio Torralba. Toward a visual concept vocabulary for gan latent space. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6784–6792, 2021.

[354] Donald A. Schön. Designing: Rules, types and worlds. *Design Studies*, 9(3):181 – 190, 1988.

[355] Prem Seetharaman, Gautham Mysore, Bryan Pardo, Paris Smaragdis, and Celso Gomes. Voiceassist: Guiding users to high-quality voice recordings. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI ’19, page 1–6, New York, NY, USA, 2019. Association for Computing Machinery.

[356] Ticha Sethapakdi and James McCann. Painting with cats: Camera-aided texture synthesis. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI ’19, page 1–9, New York, NY, USA, 2019. Association for Computing Machinery.

[357] Burr Settles and Steven Dow. Let’s get together: The formation and success of online creative collaborations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’13, page 2009–2018, New York, NY, USA, 2013. Association for Computing Machinery.

[358] Ashish Sharma, Inna W. Lin, Adam S. Miner, David C. Atkins, and Tim Althoff. Towards facilitating empathic conversations in online mental health support: A reinforcement learning approach. In *Proceedings of the Web Conference 2021*, WWW ’21, page 194–205, New York, NY, USA, 2021. Association for Computing Machinery.

[359] Rasagy Sharma and Venkatesh Rajamanickam. Using interactive data visualization to explore non-linear movie narratives. *Parsons Journal for Information Mapping*, 2013.

[360] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8242–8250, 2018.

[361] Yang Shi, Nan Cao, Xiaojuan Ma, Siji Chen, and Pei Liu. Emog: Supporting the sketching of emotional expressions for storyboarding. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI ’20, page 1–12, New York, NY, USA, 2020. Association for Computing Machinery.

[362] Evan Shimizu, Matthew Fisher, Sylvain Paris, James McCann, and Kayvon Fatahalian. Design adjectives: A framework for interactive model-guided exploration of parameterized design spaces. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST ’20, page 261–278, New York, NY, USA, 2020. Association for Computing Machinery.

[363] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online, November 2020. Association for Computational Linguistics.

[364] Ben Shneiderman. *Supporting Creativity with Advanced Information- Abundant User Interfaces*, pages 469–480. Springer London, London, 2001.

[365] Ben Shneiderman. Creativity support tools: Accelerating discovery and innovation. *Commun. ACM*, 50(12):20–32, December 2007.

[366] Ben Shneiderman and Pattie Maes. Direct manipulation vs. interface agents. *Interactions*, 4(6):42–61, November 1997.

[367] Mark Shtern, Pedro Casas, and Vassilios Tzerpos. Evaluating music mastering quality using machine learning. In *Proceedings of the 28th Annual International Conference on Computer Science and Software Engineering*, CASCON '18, page 126–135, USA, 2018. IBM Corp.

[368] Maria Shugrina, Wenjia Zhang, Fanny Chevalier, Sanja Fidler, and Karan Singh. Color builder: A direct manipulation interface for versatile color theme authoring. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery.

[369] Pao Siangliulue, Joel Chan, Steven P. Dow, and Krzysztof Z. Gajos. Ideahound: Improving large-scale collaborative ideation with crowd-powered real-time semantic modeling. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, page 609–624, New York, NY, USA, 2016. Association for Computing Machinery.

[370] Tarique Siddiqui, Paul Luh, Zesheng Wang, Karrie Karahalios, and Aditya Parameswaran. Shapesearch: A flexible and efficient system for shape-based exploration of trendlines. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, SIGMOD '20, page 51–65, New York, NY, USA, 2020. Association for Computing Machinery.

[371] Leonid Sigal, Moshe Mahler, Spencer Diaz, Kyna McIntosh, Elizabeth Carter, Timothy Richards, and Jessica Hodgins. A perceptual control space for garment simulation. *ACM Trans. Graph.*, 34(4), July 2015.

[372] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[373] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-a-video: Text-to-video generation without text-video data, 2022.

[374] Vikash Singh, Celine Latulipe, Erin Carroll, and Danielle Lottridge. The choreographer's notebook: A video annotation system for dancers and choreographers. In *Proceedings of the 8th ACM Conference on Creativity and Cognition*, Camp;C '11, page 197–206, New York, NY, USA, 2011. Association for Computing Machinery.

[375] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[376] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models, 2022.

[377] Chung Sougwen. Sougwen chung-exhibition highlights, 2021. Accessed: January, 2021.

[378] Donna Spencer. *Card sorting: Designing usable categories*. Rosenfeld Media, 2009.

[379] Angie Spoto, Natalia Oleynik, Sebastian Deterding, and Jon Hook. Library of mixed-initiative creative interfaces, 2017.

[380] Susan Leigh Star. Chapter 2 - the structure of ill-structured solutions: Boundary objects and heterogeneous distributed problem solving. In Les Gasser and Michael N. Huhns, editors, *Distributed Artificial Intelligence*, pages 37–54. Morgan Kaufmann, San Francisco (CA), 1989.

[381] Susan Leigh Star. This is not a boundary object: Reflections on the origin of a concept. *Science, Technology, & Human Values*, 35(5):601–617, 2010.

[382] Christian J. Steinmetz and Joshua D. Reiss. Steerable discovery of neural audio effects, 2021.

[383] Sarah Sterman, Evey Huang, Vivian Liu, and Eric Paulos. *Interacting with Literary Style through Computational Tools*, page 1–12. Association for Computing Machinery, New York, NY, USA, 2020.

[384] Erik Stolterman and Thandapani Selvan. Designerly tools. *Undisciplined! Design Research Society Conference*, 01 2008.

[385] Anselm L. Strauss and Juliet M. Corbin. *Basics of qualitative research: techniques and procedures for developing grounded theory*. Sage Publications, Thousand Oaks, Calif, 1998.

[386] Hendrik Strobelt, Albert Webson, Victor Sanh, Benjamin Hoover, Johanna Beyer, Hanspeter Pfister, and Alexander M. Rush. Interactive and visual prompt engineering for ad-hoc task adaptation with large language models. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2022.

[387] Qingkun Su, Wing Ho Andy Li, Jue Wang, and Hongbo Fu. Ez-sketching: Three-level optimization for error-tolerant image tracing. *ACM Trans. Graph.*, 33(4), July 2014.

[388] Hariharan Subramonyam, Wilmot Li, Eytan Adar, and Mira Dontcheva. Taketoons: Script-driven performance animation. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, UIST '18, page 663–674, New York, NY, USA, 2018. Association for Computing Machinery.

[389] Minhyang (Mia) Suh, Emily Youngblom, Michael Terry, and Carrie J Cai. Ai as social glue: Uncovering the roles of deep generative ai during social music composition. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery.

[390] Simeng Sun, Wenlong Zhao, Varun Manjunatha, Rajiv Jain, Vlad Morariu, Franck Dernoncourt, Balaji Vasan Srinivasan, and Mohit Iyyer. IGA: An intent-guided authoring assistant. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5972–5985, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[391] Reid Swanson and Andrew S. Gordon. Say anything: Using textual case-based reasoning to enable open-domain interactive storytelling. *ACM Trans. Interact. Intell. Syst.*, 2(3), September 2012.

[392] Amanda Swearngin, Chenglong Wang, Alannah Oleson, James Fogarty, and Amy J. Ko. Scout: Rapid exploration of interface layout alternatives through high-level design constraints. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[393] Haruki Takahashi and Jeeeun Kim. 3d pen + 3d printer: Exploring the role of humans and fabrication machines in creative making. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery.

[394] Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J. Martin, Animesh Mehta, Brent Harrison, and Mark O. Riedl. Controllable neural story plot generation via reward shaping. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5982–5988. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[395] Yuzuru Tanahashi and Kwan-Liu Ma. Design considerations for optimizing storyline visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2679–2688, 2012.

[396] Makarand Tapaswi, Martin Bäuml, and Rainer Stiefelhagen. Storygraphs: Visualizing character interactions as a timeline. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 827–834, 2014.

[397] Cesar Torres, Jessica Chang, Advaita Patel, and Eric Paulos. Phosphenes: Crafting resistive heaters within thermoreactive composites. In *Proceedings of the 2019 on Designing Interactive Systems Conference*, DIS '19, page 907–919, New York, NY, USA, 2019. Association for Computing Machinery.

[398] Cesar Torres, Wilmot Li, and Eric Paulos. Proxyprint: Supporting crafting practice through physical computational proxies. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*, DIS '16, page 158–169, New York, NY, USA, 2016. Association for Computing Machinery.

[399] Cesar Torres and Eric Paulos. Metamorphe: Designing expressive 3d models for digital fabrication. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*, C&C '15, page 73–82, New York, NY, USA, 2015. Association for Computing Machinery.

[400] Theophanis Tsandilas, Catherine Letondal, and Wendy E. Mackay. Mus¡i¿ink¡/i¿: Composing music through augmented drawing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, page 819–828, New York, NY, USA, 2009. Association for Computing Machinery.

[401] S.R. Turner. *MINSTREL, a Computer Model of Creativity and Storytelling*. Technical report (University of California, Los Angeles. Computer Science Department). University of California (Los Angeles). Computer Science Department, 1992.

[402] Patricia Tzortzopoulos, Rachel Cooper, Paul Chan, and Mike Kagioglou. Clients' activities at the design front-end. *Design Studies*, 27(6):657 – 683, 2006.

[403] Giuseppe Valetto, Mary Helander, Kate Ehrlich, Sunita Chulani, Mark Wegman, and Clay Williams. Using software repositories to investigate socio-technical congruence in development projects. In *Fourth International Workshop on Mining Software Repositories (MSR'07:ICSE Workshops 2007)*, pages 25–25, 2007.

[404] Dani Valevski, Matan Kalman, Yossi Matias, and Yaniv Leviathan. Unitune: Text-driven image editing by fine tuning an image generation model on a single image, 2022.

[405] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[406] Vasilis Verroios and Michael S Bernstein. Context trees: Crowdsourcing global understanding from local views. In *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014.

[407] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. Phenaki: Variable length video generation from open domain textual description, 2022.

[408] K. Vonnegut and D. Simon. *A Man Without a Country*. Seven Stories Press, 2011.

[409] James R. Wallace, Stacey D. Scott, and Carolyn G. MacGregor. Collaborative sensemaking on a digital tabletop and personal tablets: Prioritization, comparisons, and tableaux. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 3345–3354, New York, NY, USA, 2013. Association for Computing Machinery.

[410] Nick Walton. Ai dungeon 2, Dec 2019.

[411] Jing Wang, Jianlong Fu, Jinhui Tang, Zechao Li, and Tao Mei. Show, reward and tell: Automatic generation of narrative paragraph from photo stream by adversarial training. In *AAAI*, pages 7396–7403, 2018.

[412] Sheng-Yu Wang, David Bau, and Jun-Yan Zhu. Sketch your own gan, 2021.

[413] Su Wang, Greg Durrett, and Katrin Erk. Narrative interpolation for generating and understanding stories, 2020.

[414] Stephen G. Ware, R. Michael Young, Brent Harrison, and David L. Roberts. A computational model of plan-based narrative conflict at the fabula level. *IEEE Transactions on Computational Intelligence and AI in Games*, 6(3):271–288, 2014.

[415] Kento Watanabe, Yuichiroh Matsubayashi, Kentaro Inui, Tomoyasu Nakano, Satoru Fukayama, and Masataka Goto. Lyrisys: An interactive support system for writing lyrics based on topic transition. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, IUI '17, page 559–563, New York, NY, USA, 2017. Association for Computing Machinery.

[416] Martin Wattenberg. Sketching a graph to query a time-series database. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '01, page 381–382, New York, NY, USA, 2001. Association for Computing Machinery.

[417] Marc Weber, Marc Alexa, and Wolfgang Müller. Visualizing time-series on spirals. In *Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*, INFOVIS '01, page 7, USA, 2001. IEEE Computer Society.

[418] KM Weiland. *Creating Character Arcs: The Masterful Author's Guide to Uniting Story Structure, Plot, and Character Development*. Smashwords Edition, 2017.

[419] WikiArt. Wikiart dataset, 2021. Accessed: March, 2021.

[420] Thomas Wilhelm, Manuel Burghardt, and Christian Wolff. " to see or not to see"-an interactive tool for the visualization and analysis of shakespeare plays. 2013.

[421] Blake Williford, Abhay Doke, Michel Pahud, Ken Hinckley, and Tracy Hammond. Drawmyphoto: Assisting novices in drawing from photographs. In *Proceedings of the 2019 on Creativity and Cognition*, C&C '19, page 198–209, New York, NY, USA, 2019. Association for Computing Machinery.

[422] Blake Williford, Matthew Runyon, Wayne Li, Julie Linsey, and Tracy Hammond. Exploring the potential of an intelligent tutoring system for sketching fundamentals. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[423] Erik Wolf, Sara Klüber, Chris Zimmerer, Jean-Luc Lugrin, and Marc Erich Latoschik. "paint that object yellow": Multimodal interaction to enhance creativity during design tasks in vr. In *2019 International Conference on Multimodal Interaction*, ICMI '19, page 195–204, New York, NY, USA, 2019. Association for Computing Machinery.

[424] Chenfei Wu, Jian Liang, Lei Ji, Fan Yang, Yuejian Fang, Daxin Jiang, and Nan Duan. Nüwa: Visual synthesis pre-training for neural visual world creation, 2021.

[425] Ho-Hsiang Wu, Prem Seetharaman, Kundan Kumar, and Juan Pablo Bello. Wav2clip: Learning robust audio representations from clip. *arXiv preprint arXiv:2110.11499*, 2021.

[426] Hui-Yin Wu, Francesca Palù, Roberto Ranon, and Marc Christie. Thinking like a director: Film editing patterns for virtual cinematographic storytelling. *ACM Trans. Multimedia Comput. Commun. Appl.*, 14(4), October 2018.

[427] Tongshuang Wu, Michael Terry, and Carrie J Cai. Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts. *arXiv preprint arXiv:2110.01691*, 2021.

[428] Anna Xambó, Johan Pauwels, Gerard Roma, Mathieu Barthet, and György Fazekas. Jam with jamendo: Querying a large music collection by chords from a learner's perspective. In *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, AM'18, New York, NY, USA, 2018. Association for Computing Machinery.

[429] Jun Xie, Aaron Hertzmann, Wilmot Li, and Holger Winnemöller. Portraitsketch: Face sketching assistance for novices. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, page 407–417, New York, NY, USA, 2014. Association for Computing Machinery.

[430] Peng Xu, Mostofa Patwary, Mohammad Shoeybi, Raul Puri, Pascale Fung, Anima Anandkumar, and Bryan Catanzaro. MEGATRON-CNTRL: Controllable story generation with external knowledge using large-scale language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2831–2845, Online, November 2020. Association for Computational Linguistics.

[431] Junichi Yamaoka and Yasuaki Kakehi. Depend: Augmented handwriting system using ferromagnetism of a ballpoint pen. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, page 203–210, New York, NY, USA, 2013. Association for Computing Machinery.

[432] Junichi Yamaoka, Kazunori Nozawa, Shion Asada, Ryuma Niiyama, Yoshihiro Kawahara, and Yasuaki Kakehi. Accordionfab: Fabricating inflatable 3d objects by laser cutting and welding multi-layered sheets. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings*, UIST '18 Adjunct, page 160–162, New York, NY, USA, 2018. Association for Computing Machinery.

[433] Chuan Yan, John Joon Young Chung, Kiheon Yoon, Yotam Gingold, Eytan Adar, and Sungsoo Ray Hong. *FlatMagic: Improving Flat Colorization through AI-driven Design for DigitalComic Professionals*. Association for Computing Machinery, New York, NY, USA, 2022.

[434] Huiting Yang, Liangyu Chai, Qiang Wen, Shuang Zhao, Zixun Sun, and Shengfeng He. Discovering interpretable latent space directions of gans beyond binary attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12177–12185, June 2021.

[435] Humphrey Yang, Kuanren Qian, Haolin Liu, Yuxuan Yu, Jianzhe Gu, Matthew McGehee, Yongjie Jessica Zhang, and Lining Yao. Simulearn: Fast and accurate simulator to support morphing materials design and workflows. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, UIST '20, page 71–84, New York, NY, USA, 2020. Association for Computing Machinery.

[436] Qian Yang, Justin Cranshaw, Saleema Amershi, Shamsi T. Iqbal, and Jaime Teevan. *Sketching NLP: A Case Study of Exploring the Right Things To Design with Language Intelligence*, page 1–12. Association for Computing Machinery, New York, NY, USA, 2019.

[437] Lili Yao, Nanyun Peng, Weischedel Ralph, Kevin Knight, Dongyan Zhao, and Rui Yan. Plan-and-write: Towards better automatic storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*, 2019.

[438] Tom Yeh and Jeeeun Kim. *CraftML: 3D Modeling is Web Programming*, page 1–12. Association for Computing Machinery, New York, NY, USA, 2018.

[439] Ji Soo Yi, Youn ah Kang, John Stasko, and J.A. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 2007.

[440] Mehmet Ersin Yumer, Siddhartha Chaudhuri, Jessica K. Hodgins, and Levent Burak Kara. Semantic shape editing using deformation handles. *ACM Trans. Graph.*, 34(4), July 2015.

[441] Niloofar Zarei, Sharon Lynn Chu, Francis Quek, Nanjie 'Jimmy' Rao, and Sarah Anne Brown. Investigating the effects of self-avatars and story-relevant avatars on children's creative storytelling. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–11, New York, NY, USA, 2020. Association for Computing Machinery.

[442] Biqiao Zhang, Georg Essl, and Emily Mower Provost. Predicting the distribution of emotion perception: Capturing inter-rater variability. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ICMI '17, page 51–59, New York, NY, USA, 2017. Association for Computing Machinery.

[443] Enhao Zhang and Nikola Banovic. Method for exploring generative adversarial networks (gans) via automatically generated image galleries. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery.

[444] Jiayi Eris Zhang, Nicole Sultanum, Anastasia Bezerianos, and Fanny Chevalier. Dataquilt: Extracting visual elements from images to craft pictorial visualizations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[445] Y. Zhang and L. Candy. Investigating collaboration in art and technology. *CoDesign*, 2(4):239–248, 2006.

[446] Yupeng Zhang, Teng Han, Zhimin Ren, Nobuyuki Umetani, Xin Tong, Yang Liu, Takaaki Shiratori, and Xiang Cao. Bodyavatar: Creating freeform 3d avatars using first-person body gestures. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, page 387–396, New York, NY, USA, 2013. Association for Computing Machinery.

[447] Zhenpeng Zhao, Sriram Karthik Badam, Senthil Chandrasegaran, Deok Gun Park, Niklas L.E. Elmqvist, Lorraine Kisselburgh, and Karthik Ramani. Skwiki: A multimedia sketching system for collaborative creativity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, page 1235–1244, New York, NY, USA, 2014. Association for Computing Machinery.

[448] Clement Zheng, Ellen Yi-Luen Do, and Jim Budd. Joinery: Parametric joint generation for laser cut assemblies. In *Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition*, C&C '17, page 63–74, New York, NY, USA, 2017. Association for Computing Machinery.

[449] Haiyi Zhu, Robert Kraut, and Aniket Kittur. Effectiveness of shared leadership in online communities. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, CSCW '12, page 407–416, New York, NY, USA, 2012. Association for Computing Machinery.

[450] Haiyi Zhu, Robert E. Kraut, Yi-Chia Wang, and Aniket Kittur. Identifying shared leadership in wikipedia. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, page 3431–3434, New York, NY, USA, 2011. Association for Computing Machinery.

[451] Kostas Zoumpatianos, Stratos Idreos, and Themis Palpanas. Rinse: Interactive data series exploration with ads+. *Proc. VLDB Endow.*, 8(12):1912–1915, August 2015.