ROBOTIC 3D TETRIS
Comau
Capstone Final Report

By Joseph Berman

December 21, 2020

1 PROJECT SUMMARY

1.1 Objective, Scope, and Value to Sponsor

As a frontrunner in automation and innovation, Comau aims to optimize the task of packing items into containers to save time and money (note that the specific applications of this project cannot be addressed due to the NDA I signed when starting this project). Our team sought out to create a bin packing system that would pack a container with items, both reliably and optimally. The project objectives are as follows:

1. **Determine an item's size and dimensions and find the optimal placement of an item in a container,** critical to maximizing the number of items placed in a bin.
2. **Provide a user-friendly graphical interface**, offering visualization for each placement.
3. **Create a reliable and fast system**, to minimize errors, avoid item damage, and save time.

The overall scope for this project was an independent system that can be connected to existing Comau robots. The system must include software and hardware to detect incoming objects, measure their dimensions, determine their most efficient location in the target container, and visualize these placements.

Note that functionality with respect to robotic function is out of scope. We were not responsible for developing the robot and can assume functionality for picking up and placing an object of any size and weight. Instead, our program simply instructs the robot where to place the object.

1.2 Expected Deliverables

We were expected to deliver a system (Figure 1) that can successfully control an existing robot. First, our system uses a camera to capture a point cloud of the object. Then, a segmentation algorithm uses the point cloud to determine dimensions of the object. With these dimensions, a bin packing algorithm optimally computes the object placement and sends the coordinates to the user interface for visualization.

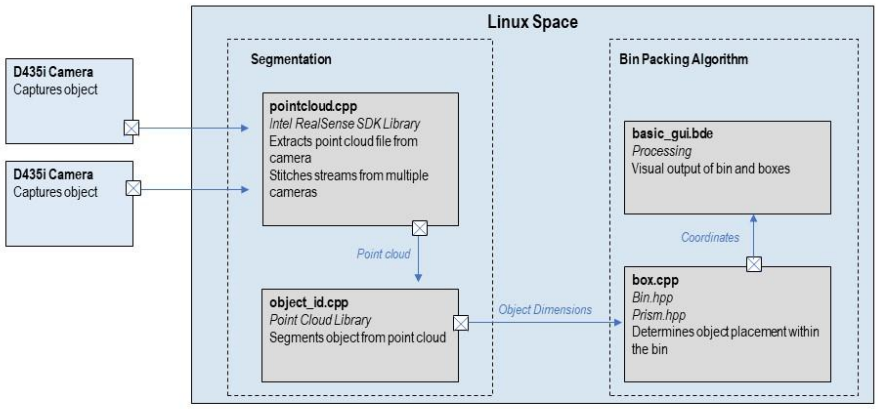

Figure 1. Core architecture of our bin packing system

In Table 1 below you will find all goals, both qualitative and quantitative, for this project. Listed with each goal is its priority level.

Goal	Priority
Fill the bin to 65% capacity	1
Fill the bin to 85% capacity	2
Place each item in under 10 seconds	1
Provide a friendly graphical user interface to illustrate current bin contents	1
Find the dimension of each item to within 2.5% margin of error	1
Write code that can interface with the robot, in turn directing it on how to place each item	3
Segment the inside of the bin to insure each item was placed in its expected location	2
Integrate all components of the system in a Rasperry Pi capable of interfacing with the Intel RealSense D435i camera	1

Table 1. List of Goals and their corresponding priorities

Goals with priority of 1 were necessary parts of our minimum viable product, whereas goals with priority of 2 represent the “nice to have” features of the final product. Finally, goals with priority of 3 represent the stretch goals, which we were to address if time permitted.

1.3 Team Approach and Solution Concepts

Hardware and Point Cloud Capture

When an object enters the field of view of the depth camera, the camera detects it and captures a point cloud of the area, as shown in Figure 2. This is done with the Intel RealSense D435i depth camera and API. The captured point cloud is then sent as input into the segmentation algorithm.

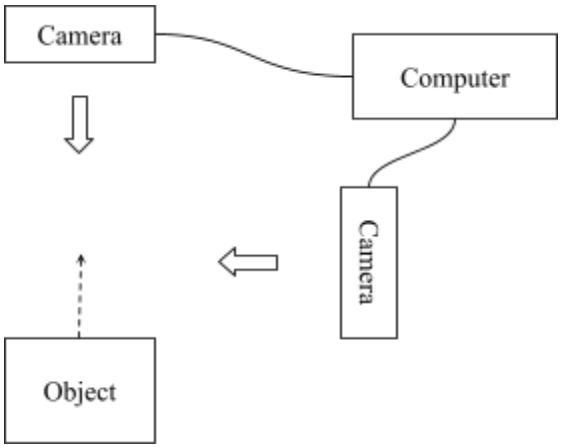

Figure 2. Setup of the camera system with respect to the object

Segmentation and Minimum Oriented Bounding Box

The goal of the segmentation algorithm is to remove all data from the point cloud given by the depth camera that is not the object of interest, and then construct a bounding box that accurately reflects the object's dimensions. To achieve this, our algorithm assumes only the object of interest is above floor level, thus removing all points at floor level. Then, by computing principal component analysis using the covariance method, we can compute the minimum oriented bounding box around an object. This approach benefits from fast performance and the ability to create accurate bounds for objects positioned at any angle relative to the camera, as seen in Figure 3.

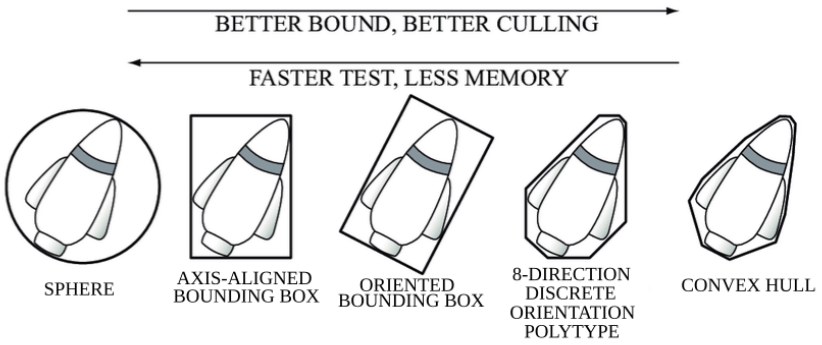

Figure 3. The correctness and performance of oriented bounding box (center) compared to other bounding volume methods [1]

Bin Packing Heuristics

For determining the optimal placement of an item in the bin, our initial algorithm design was the Online Bin Packing Heuristic to place items in series as they enter one at a time[2]. This algorithm places all items as bottom-back-left as possible by keeping track of empty maximal spaces (EMSs) that represent locations for potential item placements, as shown in Figure 4.

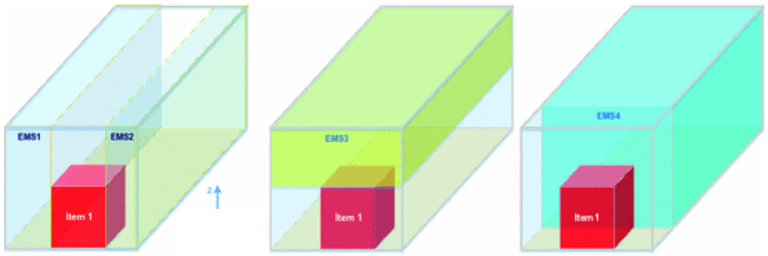

Figure 4. Four empty maximal spaces for potential placement given random item [2]

We then experimented with some algorithm adjustments. We adjusted the EMS above an item to only span across the top surface area of that item. This prevented future items from being placed in the vertical space where no actual items are underneath it. We also tried pursuing an alternative algorithm with

[1] C. Ericson, *Real-Time Collision Detection*. Cambridge, MA, USA: Morgan Kauffman, 2004.
[2] C. T. Ha, T. T. Nguyen, L. T. Bui, and R. Wang, "An Online Packing Heuristic for the Three-Dimensional Container Loading Problem in Dynamic Environments and the Physical Internet," *Applications of Evolutionary Computation Lecture Notes in Computer Science*, pp. 140–155, 2017.

non-overlapping EMS's to prevent item and EMS collisions. But in the end the overlapping EMS's algorithm that placed items based on the bottom-back-left fill variant proved to be the best performing on all test cases.

2 RESULTS

2.1 Point Cloud Generation

Our point cloud algorithm can successfully capture a point cloud of a box from multiple angles, as is shown in Figure 5 below. We also created another algorithm that is able to merge two point clouds together to create a single cohesive point cloud that it will output to the segmentation algorithm.

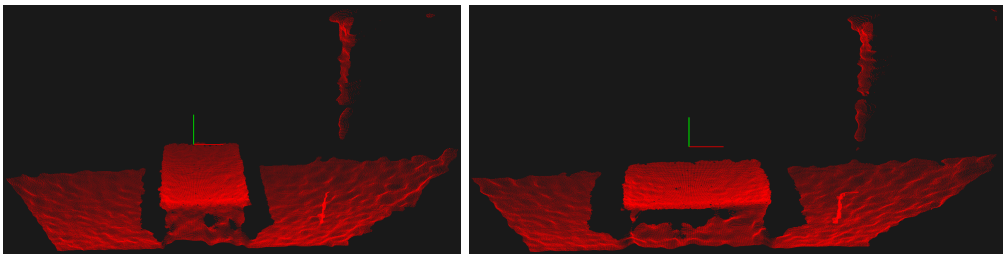

Figure 5. Point clouds of a box from two separate cameras connected to a single computer

The merged point cloud of the two point clouds shown in Figure 5 is represented below in Figure 6.

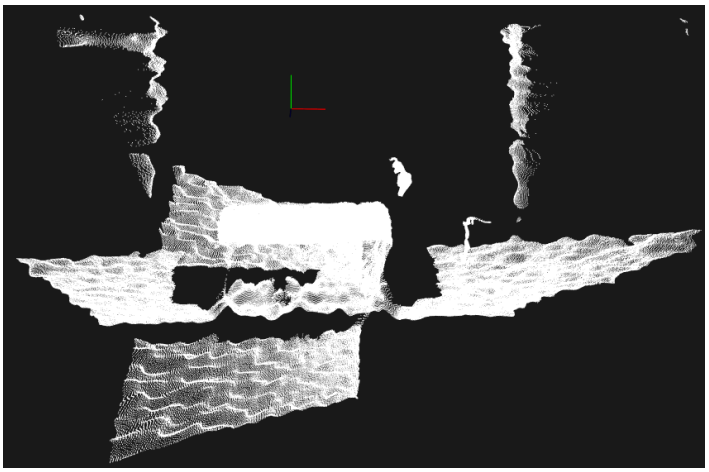

Figure 6. Merged point cloud of two point clouds created from the same box at different angles

Our point cloud algorithm worked reliably as it was accurately able to generate an precise point cloud of the item, which was then successfully transmitted to the segmentation step.

2.2 Segmentation

We implemented a working algorithm for both the segmentation and oriented bounding box computations, as shown in Figure 7 below.

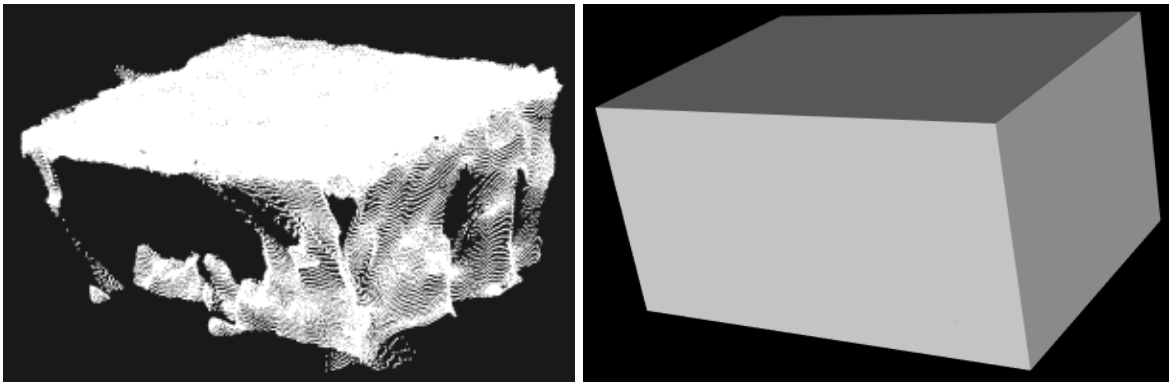

Figure 7. Merged and filtered point cloud with its minimum oriented bounding box

The aforementioned algorithm takes in the merged point clouds that were produced in the point cloud generation step of the process. Then after all ground noise is segmented out of the point cloud using basic filtering methods we are able to use the oriented bounding box method to find the dimensions of the original object. We tested our approach on a sample size of 100 boxes, in doing so we found that our algorithm can determine the dimensions of each box to within a 2% margin of error. This means that the performance of our segmentation meets one of the primary goals of the minimum viable product.

2.3 Bin Packing

In testing our bin packing algorithm we randomly generated 50 test cases, comprised of 140 boxes, each box was generated with varying lengths, widths and heights. Shown below in Figure 8 is a visualization of one of these test cases.

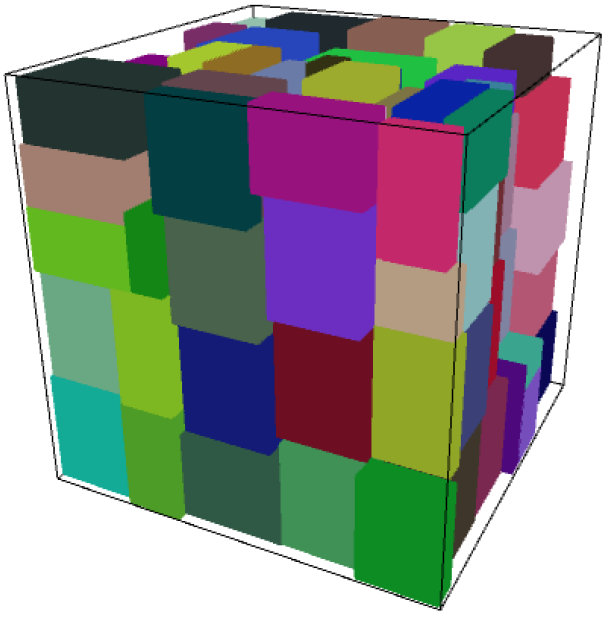

Figure 8. Results of the bin packing algorithm on one of the 50 randomly generated cases

In the test case represented in Figure 8 we were able to place 123 of the 140 items, resulting in a fill percentage of 75.328%. Our average fill percentage was approximately 73%, while the average number of items placed was 117. Therefore, we were able to achieve the minimum viable product goal of having an average fill percentage of 65% but we still fell short of the 85% fill percentage goal.

2.4 Full System Integration

The video linked here shows that we were successfully able to integrate all components into one fully functioning system. In the video you can see that each item is "placed" in the imaginary bin in under 4 seconds. Since we were unable to work directly with the robotic system, we were unable to test that the entire system can place each object in under 10 seconds, however, we are confident that the 6 seconds left for the robot to pick up and place the item will be more than sufficient based on industry knowledge.

3 CONCLUSION

3.1 Analysis of Goals

Our team was successful in delivering the minimum viable product to Comau aside for the fact that we were unable to test that the entire system can place an item in under 10 seconds. However, since our system was able to place each item within the graphical user interface in under 4 seconds we are confident that the system, once integrated with the robotic arm, will be able to place each box in the container in under 10 seconds. We unfortunately were unable to achieve any of our nice to have and stretch goals. If more time were to be allocated to this project the primary focus would be on increasing the bin packing algorithms fill capacity to achieve the goal of 85%. We would also like to integrate our system with the robotic arm to test the full capabilities of what we produced.

3.2 Handoff

At this point we have handed all of our work off to Comau so that they can continue to develop a fully functioning robotic 3-D bin packing system. To simplify the handoff process we have stored all code in one Github master branch that can be accessed by all members of the Comau team. We plan to be accessible in the event Comau would like to consult with us on any of the work we have done, but at this point our team has relinquished full control of the project to Comau for further development.