HONORS CAPSTONE REPORT

Shivastu Kartik

Abstract

Autonomous vehicles offer a wide variety of benefits for the transportation sector, including increased efficiency, lower costs and wider access to transportation. Current research at the Automotive Research Center at the University of Michigan aims to extend autonomous capabilities to off-road vehicles. Military vehicles operating in a priori unknown environments stand to realis gains in combat effectiveness and personnel safety through enhanced autonomous capabilities.

A number of approaches have been developed for path planning of autonomous vehicles, which select the vehicle's trajectory toward its objective. These algorithms must consider obstacle avoidance, path optimization under various metrics, and computational limits for embedded hardware. Reachability Based Trajectory design offers a particularly promising approach for mathematically provable safe navigation that can work with limited on-board real-time computational resources. However, existing implementations do not account for the unique dynamics of off-road vehicles, which are affected by variations of terrain, as well as tire-soil interactions. The focus of this work will be to adapt Reachability Based Trajectory Design algorithms to off-road applications. We will aim to extend the mathematically proved safety guarantees offered by Reachability Based Trajectory Design to be compatible with more complex dynamics models that capture the behavior of off-road vehicles in working in a priori unknown landscapes. We will work to factor in effects of terrain gradients, as well as tire-soil interactions to the algorithm, while ensuring feasible implementation with limited real-time processing capabilities.

Prepared By	Shivastu Kartik
Reviewed By	Professor Bogdan Epureanu

I. Introduction

The overall goal of this project is to extend effective, and safe autonomous navigation and control capabilities for off-road vehicles. This section will provide an overview of the motivations for the problem, the approach adopted, the key challenges overall and the nuances in this specific context.

Motivation

Autonomous vehicles have the potential to offer significant benefits to the transportation sector. These include increased efficiency and reduced operating costs. Further, increased intelligence of vehicle control systems offers the opportunity to take full advantage of improved digital technologies. Multiple streams of information can be made available to a connected vehicle, including

1) Onboard sensors: Such as multiple cameras, LiDAR sensors, distance sensors among others
2) Central information sources: GPS routing networks, command centers
3) Other connected vehicles

This opens the possibilities of making the driving safer, more efficient as well as more optimized based on the user's definition of optimality.

Automation of military off-road vehicles holds a multitude of benefits. The most significant of which is increased personnel safety. Automating vehicles traversing through hostile regions, especially those performing non-combat operations like reconnaissance, and resupply would reduce the number of soldiers placed in the line of fire. A study by the Rand Corporation [1] estimates that up to 78 US soldiers would be removed from danger if a single US Army convoy were to operate fully autonomously. Even if full autonomy is difficult, current research [1] into follower vehicles (a single, crewed lead vehicle, followed by multiple autonomous follower vehicles) would greatly reduce the number of troops required for frontline non-combat operations. Further autonomous capabilities would eliminate human error as a source of accidents or other incidents in military operations. This is especially true in the case of certain algorithms that offer safety guarantees.

In addition to safety, autonomous operations allow the integration of advanced digital capabilities. An autonomous connected vehicle could receive information from multiple on-board sensors (including multiple cameras, and LiDAR sensors). It would be able to connect to and receive information from multiple other sources, such as a central command center, satellite and GPS data, as well as any other information relevant to the operation. This could allow for better optimization of the vehicle's operation integrating multiple teams of data to make informed decisions better than a human driver could process.

Navigation Algorithms Overview

To develop autonomous capabilities for military vehicles, we consider the following breakdown of tasks required to guide a vehicle to its destination. This is the approach used by several studies, including the work of the ROAHM lab at the University of Michigan whose work this project is based off [2][3]. The external inputs required for this breakdown are a map of the area of operation (although there are no specific

Deleted: ¶
Deleted: offer
Deleted: S
Deleted: I
Deleted: S
Deleted: R
Deleted: etc.
Deleted: C
Deleted: V
Deleted:
Deleted: etc
Deleted: optimisation
Deleted:
Deleted: is

constraints on what information, or how much information this map should contain), and a user-specification of the destination (i.e. where does the operator want the autonomous vehicle to go)

1) **High Level Planner:** Given a destination, the high-level planner breaks down the path towards the destination into a series of way-points. These way-points act as intermediate destinations.
2) **Trajectory Planner:** The trajectory planner selects a path that vehicle should follow towards each successive way point.
3) **Low Level Planner:** The low-level planner is the one that directly interacts with the vehicle hardware. By applying actuation to the vehicle (e.g. throttle input or steering input), the low level planner makes the vehicle actually follow the path suggested by the trajectory planner.

Our work in this project focuses on the trajectory planner. The trajectory planner is the software component that interacts directly with the vehicle dynamics, as it must factor in the behavior of the vehicle in its selection of trajectories that are both feasible (i.e. can be achieved by the low level planner while adhering to actuation and dynamic limits) as well as safe (i.e. do not lead the vehicle into unsafe positions or collide with obstacles). We discuss this aspect, as well as the RTD paradigm, which we use for trajectory planning.

II. Trajectory Planning Algorithms and Reachability Based Trajectory Design

For the purposes of trajectory planning, we make use of Reachability Based Trajectory Design (RTD), developed at the University of Michigan's ROAHM Lab [4]. Reachability Based Trajectory Design describes a class of trajectory planning algorithms that make use of Reachable Sets to select safe trajectories for vehicles.

RTD makes use of a construct called the Forward Reachable Set (FRS). Given a system characterized by a set of states, $x \in \mathcal{R}^n$, the FRS describes the set of all points in state space that the system (in the present context, a vehicle) can reach in a certain finite time window.

Using the FRS, the RTD algorithms complete their path planning in a five-step-process. The paths are planned for a fixed (and usually small) time horizon, known as the planning horizon t_{plan}. The paths are planned in succession - each path is executed by the low level controller while the next one is being planned. The five step-process is as follows

1) Parametrize trajectories of the system (for example, by constant control input required to achieve it)
2) Compute the FRS for a time interval equal to t_{plan}
3) Intersect the set with the set of known obstacles. Identify a subset of the FRS excluding the points that lie in or near an obstacle (see Note A)
4) Map the safe subset from (2) to the trajectory parameter space; this yields a set of "safe" trajectory parameters that correspond to safe trajectories
5) Optimize over the remaining safe trajectories to pick the one that best suits the needs of the overall task
6) Repeat steps 3-5 for each planning window

This effectively makes RTD a form of receding horizon planning. In addition to this key algorithm, there are a few other key components of RTD based algorithms that we discuss briefly.

Deleted:
Deleted: high level
Deleted: of reachability based trajectory design
Deleted: (under Professor Ram Vasudevan)
Deleted: T
Deleted: P
Deleted: A
Deleted: Reachability Based Trajectory Design
Deleted: x¶
Deleted:
Deleted: ¶ T
Deleted: Forward Reachable Set
Deleted: D
Deleted: plath
Deleted: L
Deleted: L
Deleted: C
Formatted: Indent: Left: 0 cm
Deleted: Forward Reachable Set (
Deleted:)
Deleted: . T
Deleted: would
Deleted: .
Deleted: ¶

Note A: To ensure safety, the footprint of the obstacles can be scaled by a safety factor to ensure that the vehicle will not collide with the obstacles even in case of sensor uncertainty and/or model imperfections.

Forward Reachable Set Computation

The computation of the Forward Reachable Set is a complex problem. The FRS must be computed both accurately as well as efficiently Further, it must be guaranteed that the FRS is, in the worst case and underapproximation of the true vehicle behavior (i.e. there must be no points that the actual vehicle is able to reach that are not included in the FRS). This is important to ensure the safety guarantees. This project focuses on the computation of the FRS, and this is discussed in more detail in the subsequent sections.

Optimization over safe trajectories

The optimization method can depend on the exact application of the algorithm, and what metrics are desired to be optimized. A common approach is to define a cost variable as a function of the trajectory parameters and/or paths. The definition of the cost variable can be used to indicate to the algorithm both undesirable and desirable characteristics by associating with them high or low costs respectively. The optimization process then simply consists of finding the trajectory with the lowest value of the cost variable. Another similar approach is to define an objective function, which functions opposite to a cost. The optimization then finds the trajectory with the maximum objective function. As an example, a cost function can be used to account for the amount of actuation a trajectory requires, distance from the waypoint, and proximity to obstacles (all of which are undesirable).

Error in Reachable Sets

The Reachable Set computed in the above algorithm is likely to not be a complete representation of the vehicle's actual reachable set for a number of reasons. The main reason is that systems like road vehicles are complex dynamic systems, and high fidelity models of their behavior are complex, and higher-dimensional. This makes computing the FRS a slow process (we will discuss the reasons for thin the next section). Therefore, there exists a tradeoff between how accurate the FRS computation is and how much time and processing power it requires. Even though the FRS can be computed offline, due to practical limits, simplified, lower dimensional models are used, which leads to some error between the actual dynamics of the vehicle and ones upon which the FRS is computed.

In addition, parameter uncertainty must be taken into consideration. Models for complex systems, such as vehicles, can contain numerous parameters, whose values must be identified either by tuning or experimentation and measurement on the actual vehicle. This gives rise to uncertainty in the model parameters, which could result in even high fidelity models not being entirely accurate.

Deleted: ¶

III. Forward Reachable Sets

A rigorous definition of the Forward Reachable Set can be found in [4]. A formal definition is as follows.

Definition

Given a system characterized by a state vector $\underline{x}(t)$ and dynamics given by

$$\frac{d}{dt}\,\underline{x} = f(\underline{x}, u),$$

where u represents the control input.

Further, given a set of initial states $\mathcal{L}$ and a time horizon t_{plan}, the forward reachable set is a set $F(t_{plan})$ such that

$$F(t_{plan}) = \{x_i \mid \exists\, u, \underline{x_0}\ s.t.\, x(\bullet) = f(\underline{x}, u), \underline{x}(0) \in \mathcal{L}, \underline{x}(t_{plan}) = x_i\}.$$

The FRS therefore defines all the points in state space that the system can reach, starting from a set of initial states $\mathcal{L}$ over a finite time horizon t_{plan}.

Computation

Computation of the Forward Reachable Set is a central problem in the development of RTD based navigation algorithms. The most common method described in the literature on this subject is the Hamilton-Jacobi Reachability method.

The Hamilton-Jacobi Reachability method can be illustrated using a construct analogous to the FRS, called the Backwards Reachable Set (BRS). The BRS defines the set of states that, when propagated forward could lead the system into a given set of states $\mathcal{L}$ over a time horizon T.

The Hamilton Jacobi Method sets up the system as a two-player game. The system is taken to have dynamics as follows

$$\frac{d}{dt}x(s) = f(x(s), a(s), b(s))\ s \in [t, 0], a \in A, b \in B.$$

Where $a(s)$ and $b(s)$ are the control inputs for Player 1 and Player 2 respectively. We also denote trajectories starting from state x at time t using control $a(\bullet)$ and $b(\bullet)$ as follows:

$$\zeta(s; x, t, a(\bullet), b(\bullet)) : [t, 0] \to R^n.$$

where n is the dimension of the state space.

The backwards reachable set is therefore given by

$$G(t) = \{x \mid \exists \gamma \in \Gamma(t), \forall\, a(\bullet) \in A, \zeta(0; x, t, a(\bullet), \gamma[a]) \in G_0\} \quad \text{(A)}$$

where $\Gamma(t)$ denotes all feasible input options for Player 2, and A denotes all the possible input options for Player 1. This problem formulation defines the set of all states from which the system can be "driven" into the target set G_0 within some finite time t. Note that the setup of the two-player game is such that Player 1 (whose control input is $a(\bullet)$) will try to steer the system away from the target set, while Player 2 (whose control input is $b(\bullet)$) will try to steer the system towards the target state.

Note that in Eqn. (A), player 2's input is denoted as γ which belongs to the set of feasible strategies $\Gamma(t)$. Here, $\Gamma(t)$ is the set of all non-anticipative strategies. This means that Player 2 responds to player 1's control inputs, and cannot change her response to Player 1 until Player changes their control first. This restriction is applied only to Player 2 - hence Player 1's control input can come from A - the set of all valid control input functions. This restriction gives Player 2 the advantage of factoring in Player 1's control, which is important to establish Safety Guarantees.

A cost function is now defined over the time interval $[t, 0]$ assuming Player 1 plays control $a(\bullet)$ and Player 2 plays control $b(\bullet)$. The cost is defined as $J_T(x, a(\bullet), b(\bullet)) = g(x(0))$.

Deleted: W
Deleted: s
Deleted: Forward Reachable Set
Deleted:
Deleted: two player
Deleted: ¶
Deleted: .
Deleted: ¶
Deleted: W
Deleted: B
Deleted: R
Deleted: S
Formatted: Right
Deleted: .
Deleted: W
Deleted: In
Deleted: e
Deleted: her
Formatted: Justified
Deleted: ¶
¶

The minimum cost can be found as

$$G(t,x) = \inf_{\gamma \in \Gamma(t)} \sup_{a(\bullet) \epsilon A} J_t(x, a(\bullet), \gamma[a](\bullet))$$

where $g(x)$ is a function such that

$$x \in G_0 \Leftrightarrow g(x) \leq 0$$

The BRS is thus written as $\mathcal{G}(t) = \{x | G(t,x) \leq 0\}$, and it can be shown that the function $G(t,x)$ is the solution of the following PDE $D_t G(t,x) + H(t,x,\lambda) = 0, G(0,x) = g(x)$, where

$$H(t,x) = \max_a \min_b f(x,a,b)$$

The FRS can be analogously defined as

$$\mathcal{W}(t) = \{y: \exists \gamma \in \Gamma(t), \forall a(\bullet) \in A, \zeta(t; x, 0, a(\bullet), \gamma[a](\bullet)) = y, x \in \mathcal{G}_0\}$$

where $\mathcal{G}_0$ denotes the set of initial states from which reachability is being computed. Here instead of tracing back states x from which a destination state $y \in \mathcal{G}_0$ can be reached, we are looking to define all the destination states (each destination state denoted by y) that can be reached from an initial state $x \in \mathcal{G}_0$.

This problem can be solved very similarly to the procedure for computing the BRS. The only difference is the resulting PDE is an initial value Hamilton-Jacobi PDE instead of a final value Hamilton Jacobi PDE.

Challenges in FRS Computation

Although a popular method for computing Forward Reachable Sets, Hamilton Jacobi Reachability is not ideal for many applications. This is primarily because the Hamilton Jacobi Reachability suffers acutely from the curse of dimensionality. As the number of states (i.e. the dimension of the state space) increases, the computational complexity of the calculations involved in Hamilton Jacobi Reachability grows exponentially [5]. This is especially problematic for the application this project is based on - off road autonomous vehicles. Owing to the complex dynamics of an off-road vehicle, and its interactions with terrain (in terms of both the gradient and shape of the terrain and the interaction between the soil and the tires), the overall system could end up having several states, which would make Hamilton Jacobi Reachability impractically slow.

Another approach noted in literature is to employ the Minkowski sum of zonotopes. In this method, the Reachable Sets are represented by objects called Zonotopes. Zonotopes make for efficient computation of the FRS. However, the method assumes the linearity of the system dynamics in question. In our application, this is difficult to ensure since the dynamics of the vehicle are complex, coupled dynamics, which have multiple inputs and can be distinctly non-linear.

In addition to these challenges, our application must also account for the fact that the dynamics of off-road vehicles can be more complex than those developed for passenger vehicles. Notably, we anticipate having additional inputs (or disturbances) in the form of the interactions of the vehicle with the terrain, including both the effects of the terrain gradient and the shape of the terrain (e.g., the vehicle accelerates faster when moving downhill) as well as the effects of the interactions between the tires and the soil (e.g., the vehicle

Deleted: ¶
Deleted: W
Deleted: ¶
Deleted: ¶
¶
I
Deleted: ¶
Deleted: ¶
¶
W
Deleted: forward reachable set
Deleted: ¶
Deleted: W
Deleted: H
Deleted: offroad
Deleted: Z
Deleted: will
Deleted: sharply
Deleted: soid
Deleted:

has less traction to accelerate and turn when driving on ice than on compacted soil). Further, we must consider the dynamics of the vehicle itself - owing to the nature of the environment, we cannot assume effects like load transfer, which can be ignored for passenger vehicles in some conditions, are negligible.

IV. Proposed Solution

Overview

The proposed solution is an algorithm for computing the forward reachable set of a complex system. It aims to adapt RTD algorithms for application to off-road autonomous vehicles. Specifically, it is hoped that this method allows the integration of significantly more complex dynamics directly into the RTD framework. This will allow the algorithm to provide the mathematically proven safety guarantees that RTD promises, while considering the dynamics of the off-road vehicle, as well as the terrain that it crosses.

We also recognize the importance of ensuring that the FRS computation remains computationally feasible. Our method aims to ensure that the computational complexity of the FRS computation, despite a potentially large number of states and little opportunity for model reduction, does not grow exponentially with the number of states, as is known to happen for the Hamilton-Jacobi Method.

The method adopted will be based on differential geometry, and will consider the dynamics of the vehicle. The current aim is to account for other effects (such as terra-mechanics, and effects of slope) as disturbances. The method will also provide a modified means to compute the intersection of the Reachable Set with the set of known obstacles, thus ruling out unsafe trajectories. The current plan is for the method proposed to output a set of feasible, safe trajectories. Existing optimization methods will have to be adapted somewhat to work with the trajectories as their representation is likely to be slightly different, but the difference will not be significant. The optimal trajectory selected will then be passed on to the low level controller as specified in the RTD framework.

A few changes will likely be made to the overall framework. Most notably, it might be necessary to re-compute at least some parts of the reachable set as more information on the soil and terrain shape becomes available to the vehicle from its sensors. The exact details will depend on the amount of information the vehicle can be presumed to have beforehand, and how the soil and terrain shape affect the reachability. However, the hope is that the method proposed enables the smooth integration of complex dynamics into the RTD framework, thereby allowing an extended set of applications to take advantage of the mathematically proven safety guarantees offered by RTD algorithms.

Current State

As described earlier, an algorithm based on differential geometry has been proposed to efficiently compute the FRS for a specific situation relevant to the intended application in off road autonomous vehicles. The first step was to develop the algorithm based on a simplified vehicle model as a test case. For this purpose, we used a kinematic bicycle model. A system has also been found to correlate the FRS computation with trajectory parameters - in this steering and engine input - that can be passed to a lower level controller to execute a certain trajectory.

Deleted: will
Deleted: ve considerably
Deleted:
Deleted: offroad
Deleted: taking into account
Deleted: offroad
Deleted: take into account
Deleted:
Deleted: L
Deleted: L
Deleted: C
Deleted: M
Deleted: P
Deleted: S
Deleted: G
Deleted: Forward Reachable Set
Deleted: ARC's
Deleted: the
Deleted: K
Deleted: B
Deleted: M

The work with the kinematic bicycle model was intended as a proof of concept to demonstrate the mathematical feasibility of the proposed approach. The next step will be to implement the algorithm in code to verify its working against the simplified model used (it has been verified mathematically, this step will verify the numerical steps in the processing). Further work will also consist of expanding the formation of the algorithm. Primarily, this will be to include a more complex vehicle dynamics model, but we also hope it might be later extended to more general abstractions of complex system behavior.

Deleted: vehicle

Deleted: ¶
¶

V. Highlights of Literature Review

Deleted: : Other attempts of comparable advantage

In this section, we highlight a few notable efforts to address the challenges of computing the FRS in contexts similar to ours.

FRS Computation using Neural Networks

Paper: Learning Approximate Forward Reachable Sets Using Separating Kernels [6]

Overview of Method

The authors in this study propose a new method of estimating the reachability of systems where the dynamics are either not well known, or too complex for traditional FRS methods. Their method first identifies as identification function $L(x)\ R^n \rightarrow R$ such that, for the FRS represented by $\mathcal{G}$

$$L(x) = 1 \rightarrow x \in \mathcal{G}$$

The proposed method then sets up a neural network classifier to classify data in the form of points in the state space into whether or not they are reachable. Parameters include the initial state $\mathcal{G}_0$ and the time horizon.

Analysis for present application

The method presented does show promising results. It also holds two distinct advantages. First, it greatly reduces the computational complexity that might otherwise have been required for complex systems. This is because the specific dynamics relevant to the problem are captured by the much simpler structure of a neural network. In addition, the network is able to capture nonlinear dynamics through the structure of the network itself, without the need for model reduction. This is useful in the case of complex systems where the dynamics may not be fully understood, such as the case of the off-road vehicle.

However, this method would not directly suit our application for two key reasons. The main reason is that the neural network based approach loses the safety guarantees offered by the RTD framework, which motivated the use of the RTD Framework, and this project to adapt it for off road autonomous vehicles. There is no way, in this method, to guarantee that the FRS is a conservative approximation of the reachability of the vehicle, or that the vehicle will never enter an unsafe trajectory. This would depend on the nature and quality of the training data made available to the neural network. Given that models do currently exist to model and simulate the behavior of off-road autonomous vehicles at the ARC, this method would not provide any notable advantages. Further, the success of this method is dependent on the availability of a large amount of training data. Given that the vehicle model consists of several states coupled together with complex dynamics, the state space is quite extensive, and obtaining enough data to

Deleted: off road

train the model will be a challenging task for us. Live testing is also not a feasible source of this data since it would not be practical to collect enough data to properly train the classifier for an application like ours.

FRS Computation using efficient implementation of optimal control

Paper: A computational method for non-convex reachable sets using optimal control

Overview of Method

The authors of this study propose a new method for computing the reachability of non-convex reachable sets using optimal control.

The paper recognizes the challenges of ensuring that Hamilton Jacobi based calculations remain computationally feasible. The method proposed works on the principle of dividing the state space evenly into a grid, with each element of the grid representing a smaller subset of the state space. Then, the optimal control problem is formulated to include only the subsets. The resulting computation projects the FRS onto the grid. The aim is to solve a number of smaller optimal control problems as opposed to a large optimal control problem. This is somewhat analogous to the Divide and Conquer class of efficient algorithms that divide a large problem into smaller problems that are collectively easier to solve.

Analysis for present application

This method is also quite promising, and does represent a significant improvement over Hamilton Jacobi reachability. The method offers greater flexibility as well, since there is a tradeoff between the grid resolution, and the number of optimal control sub-problems to solve. A finer grid giver more accurate results, but includes more overhead.

However, this method would not be ideal for our application. This is because the method, being an optimal control problem, cannot guarantee globally optimal solutions. This means that there might exist optimal paths that the algorithm cannot find. It also poses a risk of non-optimal local minima confusing the algorithm resulting in a sequence of paths that do not achieve the overall goal. In addition, research along similar lines of optimization of a trajectory is currently ongoing at the ARC, and to our best knowledge, would be just as well if not better suited to our needs.

References

[1] McKay, Shawn, Matthew E. Boyer, Nahom M. Beyene, Michael Lerario, Matthew W. Lewis, Karlyn D. Stanley, Randall Steeb, Bradley Wilson, and Katheryn Giglio, *Automating Army Convoys: Technical and Tactical Risks and Opportunities*, Santa Monica, Calif.: RAND Corporation, RR-2406-A, 2020. As of May 16, 2022: https://www.rand.org/pubs/research_reports/RR2406.html

[2] S. Kousik, S. Vaskov, F. Bu, M. Johnson-Roberson, R. Vasudevan. "Bridging the Gap Between Safety and Real-Time Performance in Receding-Horizon Trajectory Design for Mobile Robots." Link

[3] S. Vaskov, S. Kousik, H. Larson, F. Bu, J. Ward, S. Worrall, M. Johnson-Roberson, R. Vasudevan. "Towards Provably Not-at-Fault Control of Autonomous Robots in Arbitrary Dynamic Environments." https://arxiv.org/abs/1902.02851

Deleted: of optimal

Deleted: recognises

Deleted: ¶

Formatted: Font: Not Bold

Formatted: Indent: Left: 0 cm, Hanging: 0.63 cm

Formatted: Font: (Default) Times New Roman, 11 pt

Formatted: Font: 11 pt

Formatted: Font: (Default) Times New Roman

Deleted: ¶

Formatted: Font: 11 pt

Deleted: ¶

Formatted: Justified, Indent: Left: 0 cm, Hanging: 0.63 cm, Space After: 0 pt

Formatted: Font: (Default) Times New Roman

Formatted: Font: 11 pt

Formatted: Indent: Left: 0 cm, Hanging: 0.63 cm, Space After: 0 pt

Formatted: Font color: Auto

Deleted: Link

Formatted: Font: (Default) Times New Roman, Font color: Custom Color(RGB(34,34,34)), Highlight

Formatted: Font: 11 pt, Font color: Custom Color(RGB(34,34,34))

[4] S. Bansal, “Introduction to reachability to share - people,” Introduction to reachability. [Online]. Available: https://smlbansal.github.io/Papers/Introduction_to_Reachability_to_Share.pdf[Accessed: 17-May-2022].

[5] Bansal, Somil, Mo Chen, Sylvia Herbert, and Claire J. Tomlin. "Hamilton-Jacobi reachability: A brief overview and recent advances." In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 2242-2253. IEEE, 2017.

[6] Thorpe, Adam J., Kendric R. Ortiz, and Meeko MK Oishi. "Learning approximate forward reachable sets using separating kernels." In *Learning for Dynamics and Control*, pp. 201-212. PMLR, 2021.

[7] Robert Baier, Mathias Gerdts. "A computational method for non-convex reachable sets using optimal control" in *Proceedings of European Control Conference 2009, August 23-26 2009* in Budapest, Hungary at 97-102.

Formatted: Font: 11 pt

Formatted: Space After: 0 pt

Formatted: Font: (Default) Times New Roman

Formatted: Font: 11 pt

Deleted: ¶

Deleted: j

Formatted: Font: (Default) Times New Roman, 11 pt

Formatted: Font: (Default) Times New Roman, 11 pt

Formatted: Indent: Left: 0 cm, Hanging: 0.63 cm, Space After: 0 pt

Formatted: Indent: Left: 0 cm, Hanging: 0.63 cm

Deleted: ¶

¶

Formatted: Font: (Default) Times New Roman, 11 pt

Formatted: Font: (Default) Times New Roman