Supervised Learning Project

Report

REACHABILITY BASED SAFETY CONTROLLER FOR OVERTAKING MANUVEURS BY AVs

Supervisor:
Prof. Arpita Sinha

By:
Prachit Gupta
210100111
UG 3rd Year
Mechanical Engineering

.

ABSTRACT

Autonomous vehicles have very complicated maneuvers requiring special techniques for their control and motion planning. Overtaking particularly presents a great challenge as it involves multiple complex movements like lane changing, lane following, acceleration, and braking. This makes ensuring safety in such a complex maneuver of paramount importance. Backward reachability analysis is a standard method for guaranteeing safe maneuvers assuming worst-case actions of humans in robot-human interactions such as slow overtaking but incorporating such safety monitors while ensuring robotic planners have human-like fluidity in taking actions has always been a challenge. In this project, we formally design a robotic planner that is aware of the HJ-reachability-based safety monitor. This makes our autonomous driving stack not only collision-free against all possible actions of leading vehicles but also ensures planner performance is minimally intervened by the same. We formulate the overtaking maneuver as a multi-stage optimization problem and show that by integrating safety assurance within the model predictive controller-based trajectory planner, the vehicle is able to overtake a slow-moving leading vehicle while accounting for worst case, careless actions by the human driver and minimally deviates from planned trajectory while doing so.

MOTIVATION

Uncertainty in the predicted motion makes the automated overtaking problem hard to solve due to feasibility issues that arise from the fact that the overtaken vehicle (e.g., a vehicle driven by an aggressive driver) may accelerate to prevent the overtaking maneuver. Overtaking though challenging, is a prominent maneuver, and ensuring Safety in it is of Paramount Importance:

- Hamilton-Jacobi (HJ) reachability analysis is an important formal verification method for guaranteeing performance and safety properties of dynamical systems
- Though there has been sufficient work in ensuring safety, the conventional methods resort to switching controller strategy which causes major deviations from desired trajectories
- The autonomous vehicle should be able to avoid a collision even when the other car defies the planner's expectations and takes dangerous actions, either carelessly or with the intent to collide
- It is desirable for all this to take place while deviating minimally from planned motion while ensuring safety

LITERATURE REVIEW

HJI reachability has proved to be a reliable and robust method for guaranteeing safety against all possible anticipation actions by other drivers. Its advantages include compatibility with general nonlinear system dynamics, formal treatment of bounded disturbances, and the availability of well-developed numerical tools have been experimentally verified in [1]. While existing autonomous driving stacks tend to track the desired trajectory planned by the high-level planners, they have used a switching controller strategy to incorporate reachability safety. However in [2] authors have successfully leveraged reachability analysis to construct a real-time (100 Hz) controller that serves the dual role of (i) tracking an input trajectory from a higher-level planning algorithm using model predictive control, and (ii) assuring safety by maintaining the availability of a collision-free escape maneuver as a persistent constraint regardless of whatever future actions the other car takes. They are using precomputed value functions and have added additional constraints on the control input which is activated only when the value function becomes less than some threshold value indicating a safety breach. MPC has proved to be one of the industrial techniques for performing overtaking maneuvers as shown

in survey [3] because of its ability to make real-time predictions and handling large number of constraints. There doesn't exist a formulation as per my knowledge which uses a HJI PDE-based model predictive controller that deals with both trajectory planning and safe tracking ensuring overtaking is performed but only when it is completely safe

1. Novel Contributions

In this section, I outline the novel contributions of our approach to trajectory generation and control in overtaking scenarios.

1.1. Safe and Feasible Overtaking

This method ensures that overtaking maneuvers are only performed when they are both safe and feasible. By computing BRTS as a zero sublevel value function of a zero-sum differential game between 2 players, we can have some notion of a safety monitor ensuring safety against all possible motions by the human. The novelty here is that overtaking is not enforced, robot overtakes lead if and only if the maneuver is both safe and feasible

1.2. Reachability-based Trajectory Generation over Tracking

Unlike traditional approaches that rely on predefined reference trajectories, our system automatically generates safe trajectories based on the dynamic characteristics of the ego vehicle, and the lead vehicle. This adaptive trajectory planning enables smooth and efficient overtaking maneuvers while ensuring safety at all times.

1.3. Minimally Invasive Control Intervention

Our control strategy ensures minimal intervention to achieve safe and smooth motion during overtaking maneuvers. Unlike traditional switching controller-based approaches that abruptly switch to emergency control mode, safety is the basic characteristic of our controller strategy rather than something that has to be enforced

2. Preliminaries

2.1. Zero Level Differential Games

In differential games, the zero-level game refers to situations where the outcome is determined by whether the system reaches a given configuration under specified constraints within a certain duration. In the context of reachability analysis, the zero-level game is utilized to compute backward reachable sets (BRS), which represent the set of states from which the system can be driven to a target set within a defined time horizon.

2.2. Backward Reachable Sets (BRS)

The backward reachable set is a fundamental concept in reachability analysis. It represents the set of states from which the system trajectories can reach a specified target set despite the best efforts to avoid it. In safety-critical scenarios, such as collision avoidance protocols, the BRS identifies potentially unsafe states that should be avoided to ensure system safety.

2.3. Dynamic Programming

Dynamic programming is a key technique used to solve differential games and compute backward reachable sets. By formulating the problem as an optimization task over a cost function, dynamic programming algorithms iteratively compute the value function, which represents the expected accumulated cost over system trajectories. The Hamilton-Jacobi-Isaacs (HJI) partial differential equation (PDE) plays a central role in dynamic programming-based approaches for reachability analysis.

2.4. Bang-Bang Control

In the context of safety-critical systems, such as autonomous vehicles, the notion of bang-bang control refers to control actions that alternate abruptly between extreme values in response to changing conditions. In reachability analysis, bang-bang control strategies may be employed to keep the system trajectories within safe regions or avoid unsafe states.

3. HJI Formulation

In this work, we use reachability analysis to compute a backward reachable tube (BRT), $V(\tau)$, given an unsafe set of states L (e.g., all states where the human and robot are in collision).
Intuitively, $V(\tau)$ is the set of states from which if system trajectories start, they are guaranteed to enter into the unsafe set of states within a time horizon of τ despite the robot's best effort to avoid the unsafe set.
Let the dynamics of the human-robot system evolve via $\dot{x} = f(x, u_H, u_R)$ where f is assumed to be uniformly continuous in time and Lipschitz continuous in x for fixed u_H and u_R. Here, $x := [x_H, x_R]^T \in X$ is the joint state of the human and robot, and $u_H \in U_H$ and $u_R \in U_R$ are the human's and robot's inputs, respectively. We also assume that the state of both agents are known precisely for the entire time horizon

To ensure robustness to the possible—including worst-case—behaviors of the human agent, the computation of the BRT is formulated as a zero-sum differential game between the robot and human. The optimal value of this game can be obtained by solving the final value Hamilton-Jacobi-Isaacs partial differential equation via dynamic programming:

$$\frac{\partial V(x,\tau)}{\partial \tau} + H(x, \tau, \nabla V(x,\tau)),\ \{\lambda(x) - V(x,\tau)\} = 0, \quad \frac{\partial V(x,0)}{\partial \tau} = \lambda(x), \quad \tau \in [-T, 0],$$

where $\nabla V(x,\tau)$ is the spatial derivative of the value function and $\lambda(x)$ is the implicit surface function encoding the set of unsafe states: $L = \{x : \lambda(x) \leq 0\}$. The Hamiltonian H encodes the effect of the dynamics, robot, and human control on the resulting value and is defined as:

$$H(x, \tau, \nabla V(x,\tau)) = \max_{u_R \in U_R} \min_{u_H \in U_H} \nabla V(x,\tau)^T f(x, u_H, u_R).$$

computing the value function $V(x,\tau)$ backwards in time over $\tau \in [-T, 0]$, we can obtain the BRT at any time τ by looking at the sub-zero level set of the value function:

$$V(\tau) := \{x : V(x,\tau) \leq 0\}.$$

This encodes the set of initial (joint) states from which there does not exist a dynamically feasible safety control for the robot to perform to avoid the human. HJ reachability also synthesizes the robot's optimal safety-preserving control:

$$u_R^*(x,\tau) = \arg \max_{u_R \in U_R} \min_{u_H \in U_H} \nabla V(x,\tau)^T f(x, u_H, u_R),$$

which can be used in a least-restrictive fashion by only being applied at the boundary of the unsafe set.

4. Value Function Computation

We use the Level set method for numerically solving HJI Partial Differential Equation and model robot and lead vehicle interaction as a zero sublevel differential game of degree.
Let the dynamics of the human-robot system evolve via $\dot{x} = f(x, u_H, u_R)$ where f is assumed to be uniformly continuous in time and Lipschitz continuous in x for fixed u_H and u_R. Here, $x := [x_H, x_R]^T \in X$ is the joint state of the human and robot, and $u_H \in U_H$ and $u_R \in U_R$ are the human's and robot's inputs, respectively. We also assume that the state of both agents can be accurately sensed at all times.

To ensure robustness to the possible—including worst-case—behaviors of the human agent, the computation of the BRT is formulated as a zero-sum differential game between the robot and human. The optimal value of this game can be obtained by solving the final value Hamilton-Jacobi-Isaacs partial differential equation via dynamic programming as follows
Using The final value function V (x rel , 0) = l(x rel), we can solve the The following HJ equation is used to obtain $V(x_{\text{rel}}, t)$ whose zero sub-level set represents the BRT:

$$\text{mín}\left\{\frac{\partial V(x_{\text{rel}}, t)}{\partial t} + H(x_{\text{rel}}, \nabla V(x_{\text{rel}}, t)), V(x_{\text{rel}}, 0) - V(x_{\text{rel}}, t)\right\} = 0, \quad t \in [-T, 0] \tag{1}$$

By discretizing the state space into grids, Equation (1) can be solved via dynamic programming with the level set method. There are several numerical toolboxes based on level set methods readily available which are compared in the image below:

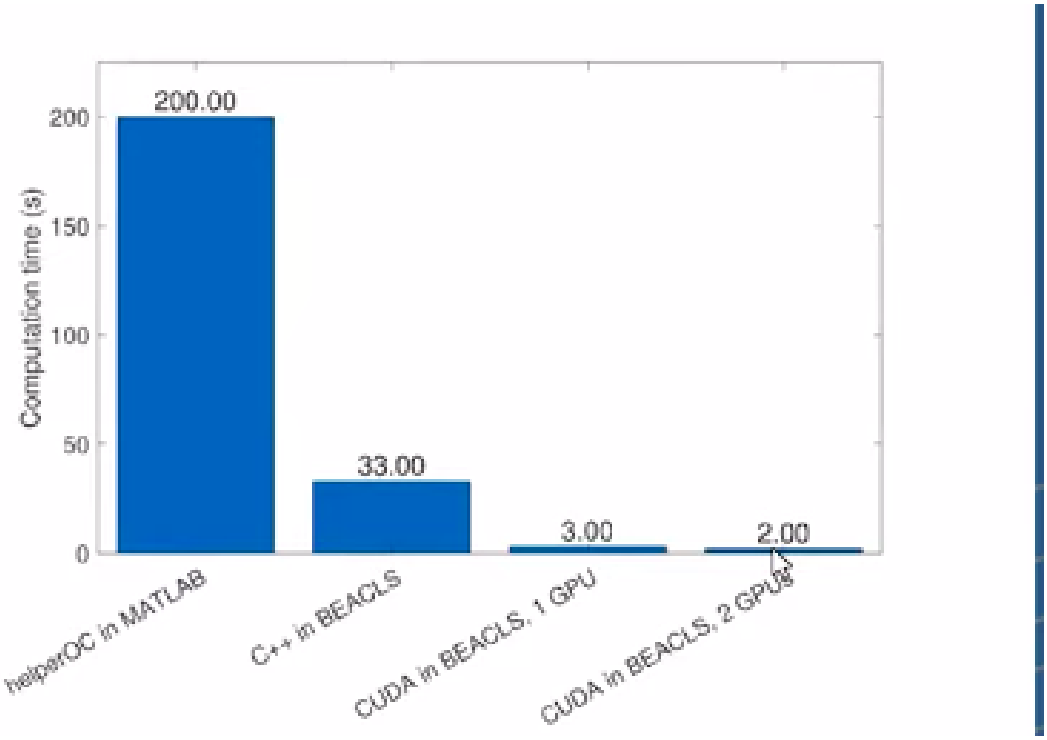


Figura 1: Comparision

5. Robot Dynamics

The dynamics of the robot are described by the following equations:

$$\begin{aligned}
\dot{x} &= v\cos(\psi+\beta) \\
\dot{y} &= v\sin(\psi+\beta) \\
\dot{\psi} &= \frac{v}{l_r}\sin(\beta) \\
\dot{v} &= a \\
\beta &= \tan^{-1}\left(\frac{l_r}{l_f}\tan(\delta_f)\right)
\end{aligned}$$

where x and y are the coordinates of the center of mass in an inertial frame (X, Y). ψ is the inertial heading, and v is the speed of the vehicle. l_f and l_r represent the distance from the center of mass of the vehicle to the front and rear axles, respectively. β is the angle of the current velocity of the center of mass with respect to the longitudinal axis of the car. a is the acceleration of the center of mass in the same direction as the velocity. The control inputs are the front and rear steering angles δ_f, and a. Since in most vehicles, the rear wheels cannot be steered, we assume $\delta_r = 0$.

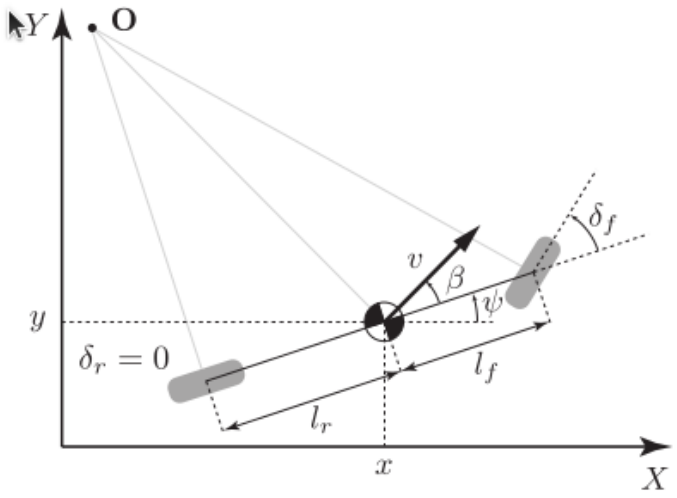


Fig. 1: Kinematic Bicycle Model

Figura 2: Illustration of Robot Dynamics

6. Human Dynamics

The human car (denoted by subscript H) is modeled using the dynamically extended unicycle model as illustrated in Figure 6. Let (p_x^H, p_y^H) be the position of the center of the human car's rear axle defined in an inertial reference frame, and c_H be the yaw angle (heading) of the human car relative to the horizontal axis. The velocity of the human car in the vehicle frame is v_H. The state for the dynamically extended unicycle model is defined as:

$$x_H = \begin{bmatrix} p_x^H \\ p_y^H \\ c_H \\ v_H \end{bmatrix}$$

The control input $u_H = \begin{bmatrix} v \\ a \end{bmatrix}$ consists of the yaw rate v and longitudinal acceleration a. The control limits of the human car are chosen such that the robot and human car share the same power, steering, and friction limits. The equations of motion for the human car are given by:

$$\dot{x}_H = \begin{bmatrix} v_H \cos c_H \\ v_H \sin c_H \\ v \\ a \end{bmatrix}$$

Owing to its simpler dynamics representation, the human car has a transient advantage in control authority over the robot car (it may change its path curvature discontinuously, while the robot may not), but by equating the steady-state control limits we ensure that the infinite time horizon BRS computation converges.

7. Relative Dynamics Between Human-Driven Car and Robot Car

In our simulation plots and animations, we consider a dynamical system that encodes the relative dynamics between the robot car and the human-driven car in a pairwise interaction. The human-driven car is modeled as an extended unicycle model, and the robot car is modeled with a high-fidelity bicycle. The state $\mathbf{x}_{\text{rel}}$ of the relative system is defined as:

$$\mathbf{x}_{\text{rel}} = \begin{bmatrix} p_{\text{rel},x} \\ p_{\text{rel},y} \\ \psi_{\text{rel}} \\ v_R \\ v_H \end{bmatrix}$$

where $p_{\text{rel},x}$ and $p_{\text{rel},y}$ are the x and y coordinates of the human-driven car in the coordinate frame centered at the geometric center of the robot car with the x-axis aligned with the heading of the self-driving car, ψ_{rel} denotes the relative heading between the two cars, and v_R (resp., v_H) denotes the speed of the robot car (resp., human-driven car).

Let $\mathbf{u}_R = [a_R, \delta f]^T$ be the robot's control input, where a_R is the acceleration and δf is the front wheel rotation, and $\mathbf{u}_H = [a_H, \omega_H]^T$ denote the human's control input, where a_H is the acceleration and ω_H is angular speed.

The evolution of the relative system is governed by the following differential equation:

$$\dot{\mathbf{x}}_{\text{rel}} = \mathbf{f}_{\text{rel}}(\mathbf{x}_{\text{rel}}, \mathbf{u}_R, \mathbf{u}_H)$$

where:

$$\begin{aligned}
\dot{p}_{\text{rel},x} &= Rl_r \sin(\beta_r) + v_H \cos(\psi_{\text{rel}}) - v_R \cos(\beta_r) \\
\dot{p}_{\text{rel},y} &= -Rl_r \sin(\beta_r) + v_H \sin(\psi_{\text{rel}}) - v_R \sin(\beta_r) \\
\dot{\psi}_{\text{rel}} &= \omega_H - \frac{v}{l_r} R \sin(\beta_r) \\
\dot{v}_R &= a_R \\
\dot{v}_H &= a_H
\end{aligned}$$

Here, l_f (resp., l_r) denotes the front (resp., rear) axle length of the robot car, and β_r is computed via $\beta_r = \tan^{-1}\left(\frac{l_r + l\tan(\delta f)}{l_r}\right)$.

8. Results

The target set is defined as the set of states where the robot and human are within a rectangle of 6×4 meters, i.e., x_{rel} and y_{rel} are constrained in a rectangle about the origin.

The control action is assumed to be bang-bang

8.1. Value Functions

Using helper Oc toolbox and notions of relative dynamics and bang-bang control sequence given above, the value functions were computed as a 5d array whose 2d slices are illustrated below:7.

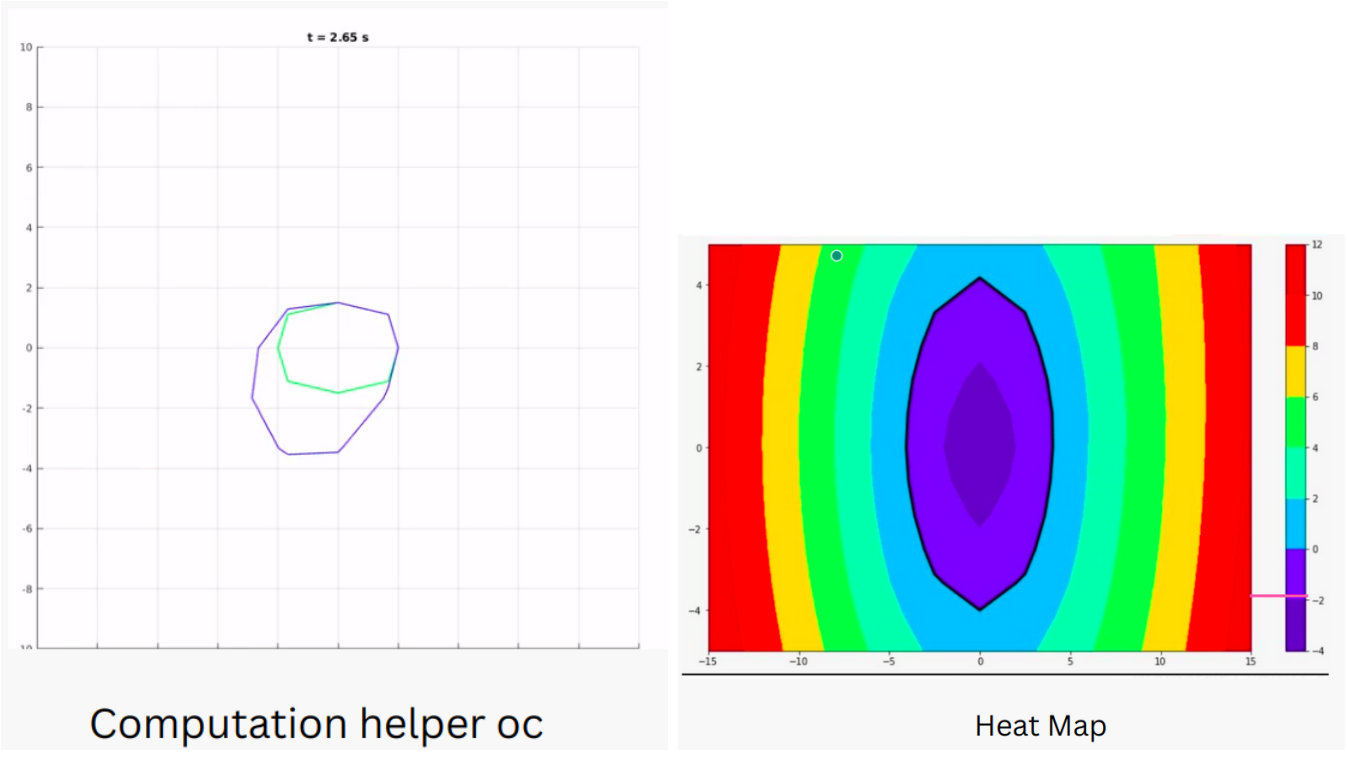
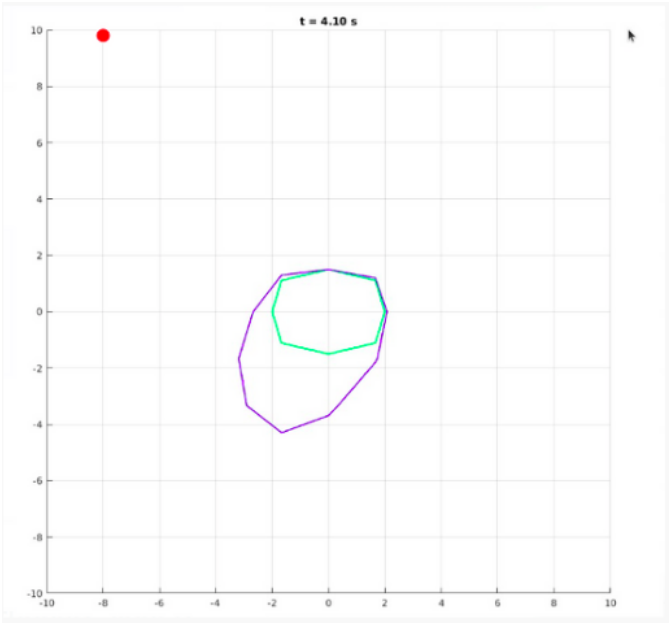


Figura 3: Results of Experiment 1

As seen in Figure HeatMap the black outlined region encompasses the BRS with all points within it having values less than 1

8.2. Validating our Computed BRTS

Extracted the 2d slice of BRT by fixing the other three states and compared results with [4] as 4.



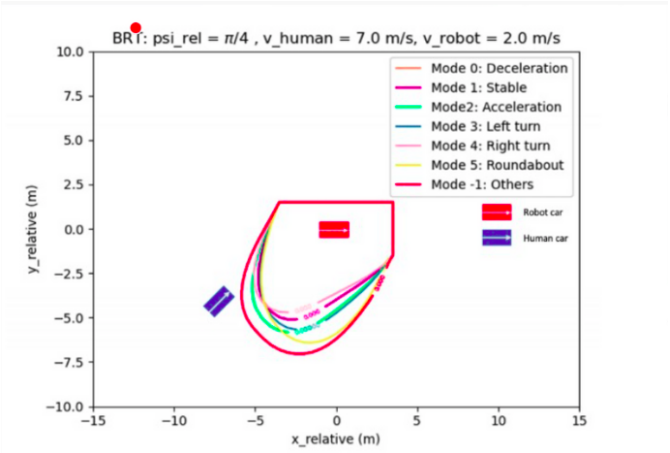


Figura 4: Results of Experiment 2

9. MPC Formulation with Kinematic Bicycle Model (KBM) Dynamics

9.1. Objective Function

The objective function aims to minimize the cost over a finite prediction horizon. It is defined as follows:

$$\min_{U,X} \sum_{k=0}^{N} \left[(X_k - X_f)^T R (X_k - X_f) + U_k^T Q U_k \right]$$

Where:

- X_k is the state vector at time step k.
- U_k is the control vector at time step k.
- X_f is the desired final state.
- R and Q are the weighting matrices for the state and control variables, respectively.

9.2. Constraints

The constraints ensure that the system satisfies dynamics, state bounds, and path constraints. These can be written as:

9.2.1. Initial Condition

$$X_0 - X_{\text{initial}} = 0$$

9.2.2. Dynamics Constraint (KBM)

$$X_{k+1} - (f(X_k, U_k) \cdot T + X_k) = 0, \quad \forall k \in [0, N-1]$$

9.3. Decision Variables

- U represents the control inputs over the prediction horizon.
- X represents the state trajectories over the prediction horizon.

9.4. Parameters

- X_{initial} is the initial state.

9.5. Summary

The MPC problem seeks to minimize a cost function while satisfying system dynamics and constraints. It optimizes control inputs over a finite prediction horizon to achieve desired state trajectories. The objective is to find an optimal control policy that minimizes the distance between the initial state and goal until the point of convergence while preserving smoothness and robustness

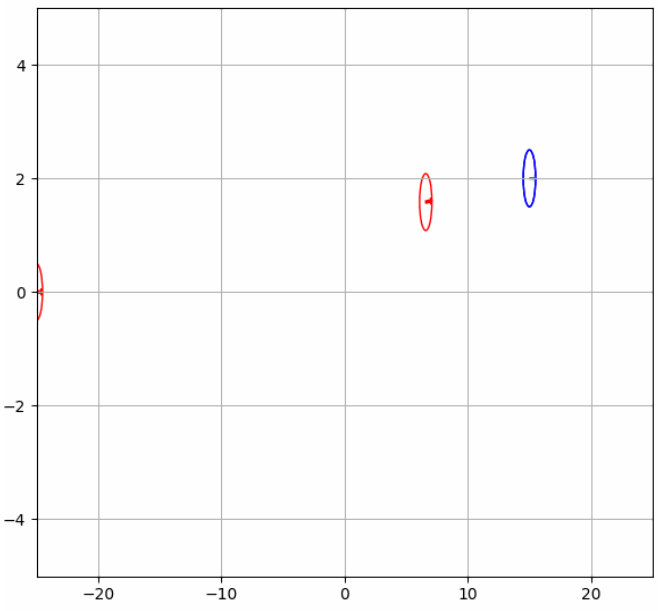


Figura 5: Visualization

10. MPC Formulation

10.1. Objective Function

The objective function aims to minimize the cost over a finite prediction horizon. It is defined as follows:

$$\min_{U,X} \sum_{k=0}^{N} \left[(X_k - X_f)^T R (X_k - X_f) + U_k^T Q U_k + \text{value}(X_k) \right]$$

Where:

- X_k is the state vector at time step k.
- U_k is the control vector at time step k.
- X_f is the desired final state.
- R and Q are the weighting matrices for the state and control variables, respectively.
- value($X_r el$) is the value function representing the deviation from the reference trajectory.

10.2. Constraints

The constraints ensure that the system satisfies dynamics, state bounds, and path constraints. These can be written as:

10.2.1. Initial Condition

$$X_0 - X_{\text{initial}} = 0$$

10.2.2. Value Function Constraint

$$\text{value}(X_r el) - \text{toleranceValue} \geq 0, \quad \forall k \in [0, N]$$

10.2.3. Dynamics Constraint

$$X_{k+1} - (f(X_k, U_k) \cdot T + X_k) = 0, \quad \forall k \in [0, N-1]$$

10.3. Decision Variables

- U represents the control inputs over the prediction horizon.
- X represents the state trajectories over the prediction horizon.

10.4. Parameters

- X_{initial} and X_f are the initial and final states, respectively.
- toleranceValue is a tolerance value for the value function.

Model predictive controller or MPC is known for its ability to handle large constraints and control inputs by solving constrained finite time-horizon optimal control problems. However, formulation of the problem itself requires local linearization and discretization, something which necessitates a need for a single global coordinate system representation. Here is where this stable embedding technique comes to the rescue which allows one to not only extend the dynamics to ambient Euclidian space for Linearizatin and Discretization steps of MPC but also do it in such a way that the manifold M becomes an invariant attractor of the modified dynamics defined in R^n.

11. Simulation Parameters

11.1. Value Function Computation

- grid size of $13 \times 13 \times 9 \times 9 \times 9$ for our 5D system
- uniformly spaced over $(p_x^{rel}, p_y^{rel}, yaw^{rel}, V_R, v_H$ in the ranges $[-10, 10] \times [-5, 5] \times [-pi/2, pi/2] \times [0, 12] \times [0, 12]$
- computing the BRS with this system and discretization takes approximately an hour on a Intel i5 CPU

11.2. MPC formulation

Cuadro 1: Constraints

Parameter	**Range**
x	[-inf, inf]
y	[-5m,5m]
phi	[-180°,180°]
v	[0 m/s ,20 m/s]
a	[-4 m/s^2,4 m/s^2]
delF	[-pi/4 rad/s, pi/4 rad/s]
Tolerance	0.2
TimeStep	0.2s
lf	1.105
lr	1.738

12. Results

12.1. MPC without safety monitor

Using CASADI solver for optimization and modeling , the following results were obtained :

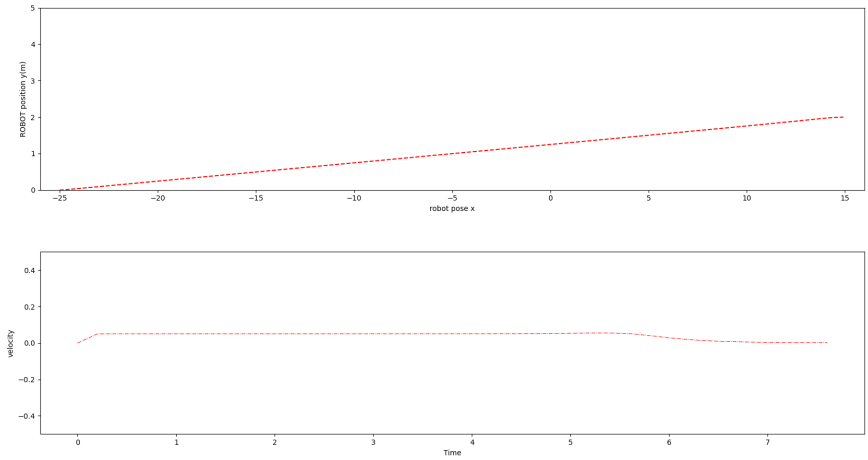


Figura 6: Trajectory

As seen in Figure we obtained smooth variation of velocity with time and smooth robust trajectories ensuring that the robot reaches the desired state

12.2. MPC with safety monitor

Using the CASADI solver for optimization and modeling, the following results were obtained :

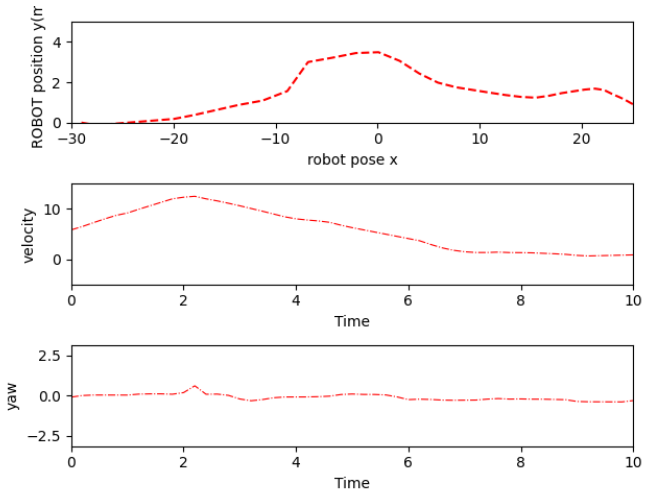


Figura 7: Trajectory

As seen in Figure the robot overtakes the lead vehicle by simultaneously generating and tracking a trajectory. Since the trajectory generation itself encompasses the notion of the value function, safety is automatically ensured while performing overtaking. The trajectories is consistently smooth except the final zone where there's a potential conflict between minimizing desired state and value Function something to be addressed later

13. Research Gaps and Future Directions

13.1. Conservative Backward Reachable Tube Synthesis (BRTS)

The existing approach to BRTS tends to be overly conservative, resulting in trajectories that include jerky deviations even when the environment is completely safe.

13.2. Proposed Solutions in Literature

1. Some literature suggests incorporating game-theoretic models of human behavior to generate BRTS that correspond only to predicted human motions, rather than considering all possible motions.

2. Another proposed solution involves creating multiple BRTS, each considering a separate action by humans, and then dynamically switching between them online.

13.3. Weak Notion of Target Set

The current notion of the target set, based solely on distance, may lead to situations where collisions are considered inevitable even when vehicles are significantly apart.

13.4. Open Research Problems

The following are open research problems in the field:

- Develop methods to mitigate the overly conservative nature of BRTS while ensuring safety.
- Investigate approaches to refine the definition of the target set to account for dynamic environmental factors.
- Explore novel techniques for integrating game-theoretic models into BRTS to better capture human behavior.

14. Software Contributions

- Developed a VALUE-FUNCTION-GENERATOR package based on helperOC with documentation on:
  - Generating value function with custom relative 5d state space
  - Generating a `CASADI` based lookup-table from the value functions
  - Scripts for integrating value functions in any MPC-based control stack
  - Scripts for animation and visualization

15. References

[1] Hamilton-Jacobi Reachability: A Brief Overview and Recent Advances Somil Bansal*, Mo Chen*, Sylvia Herbert* and Claire J. Tomlin

[2] On infusing reachability-based safety assurance within planning frameworks for human–robot vehicle interactions

[**3**] Autonomous vehicular overtaking maneuver: A survey and taxonomy Shikhar Singh Lodhi, Neetesh Kumar , Pradumn Kumar Pandey

[**4**] Prediction-Based Reachability for Collision Avoidance in Autonomous Driving Anjian Li 1 , Liting Sun 2 , Wei Zhan 2 , Masayoshi Tomizuka 2 and Mo Chen 1

[**5**] Safety Assurances for Human-Robot Interaction via Confidence-aware Game-theoretic Human Models Ran Tian , Liting Sun , Andrea Bajcsy , Masayoshi Tomizuka, and Anca D. Dragan

[**6**] Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design Jason Kong 1 , Mark Pfeiffer 2 , Georg Schildbach 1 , Francesco Borrelli