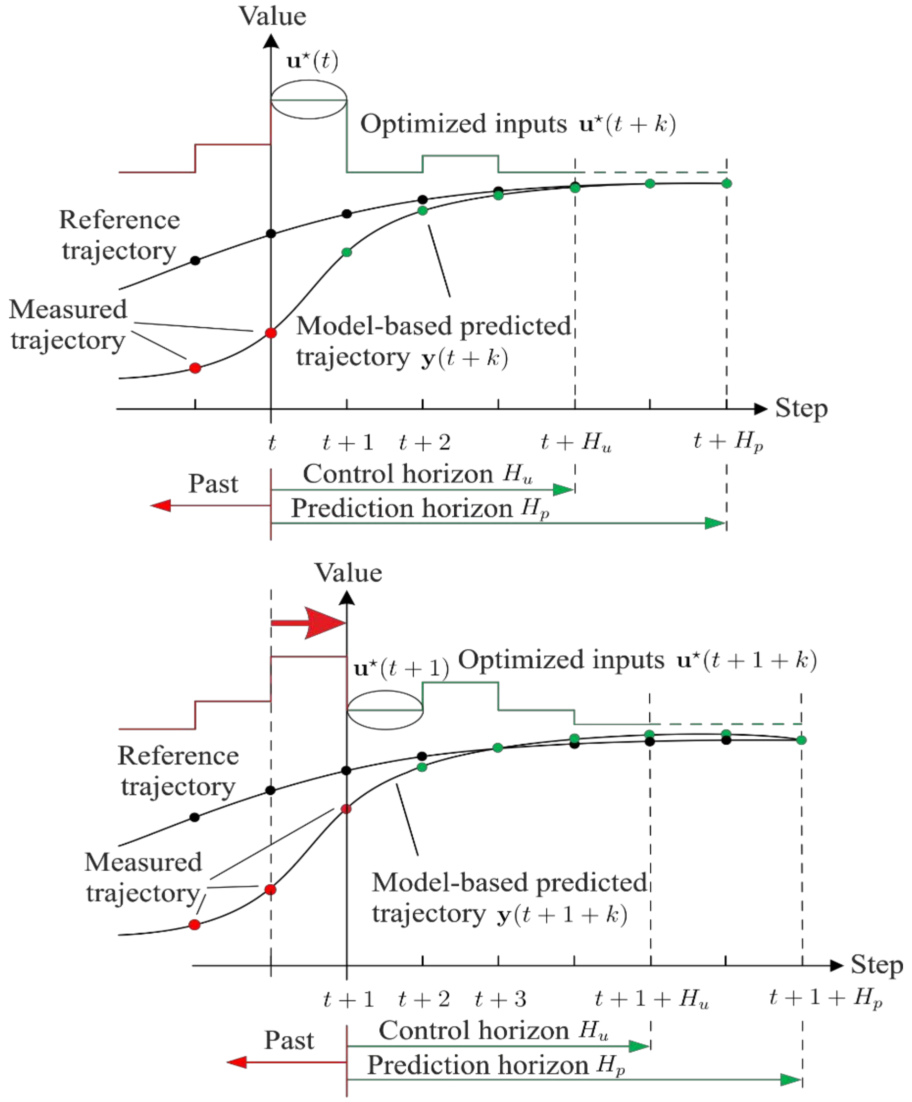
Processing of Dynamic Object Information in MPC-Planner

Mohammed Azharudeen Farook Deen | Group 9 | Mat: 429189
RWTH Aachen University I ACDC Research Project

Model Predictive Control



Numerical optimization problem

$$\min_{u_0,\ldots,u_{N-1}} \sum_{k=0}^{N-1} \|y_k - r(t)\|_2^2 + \rho\|u_k - u_r(t)\|_2^2$$

s.t. $x_{k+1} = f(x_k, u_k)$ prediction model

$y_k = g(x_k)$

$u_{\min} \leq u_k \leq u_{\max}$ constraints

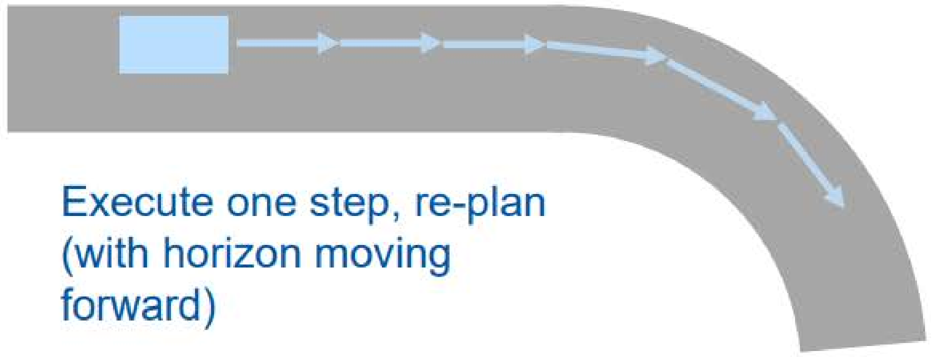
$y_{\min} \leq y_k \leq y_{\max}$

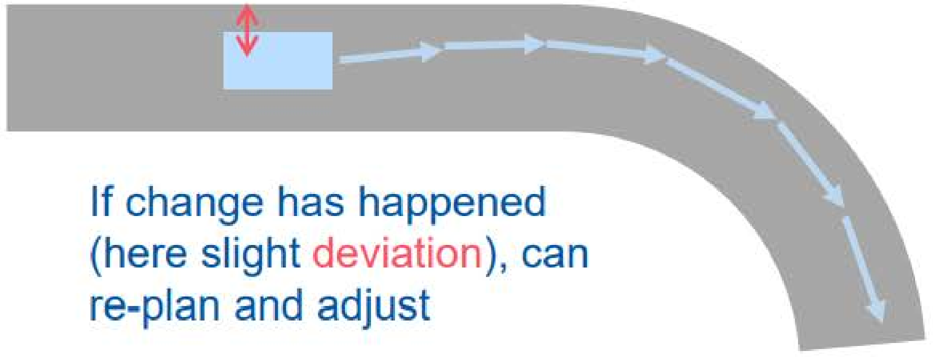
$x_0 = x(t)$ state feedback

Receding Horizon

Receding horizon: Plan for a horizon, execute one step, re-plan (i.e., move the horizon further), etc.

- Apply only the first input.
- Observe the next state.
- Restart the optimization with new state as initial condition. That is: We re-solve the optimization problem at every time step t.
- Keep going forever, or until the task is solved.

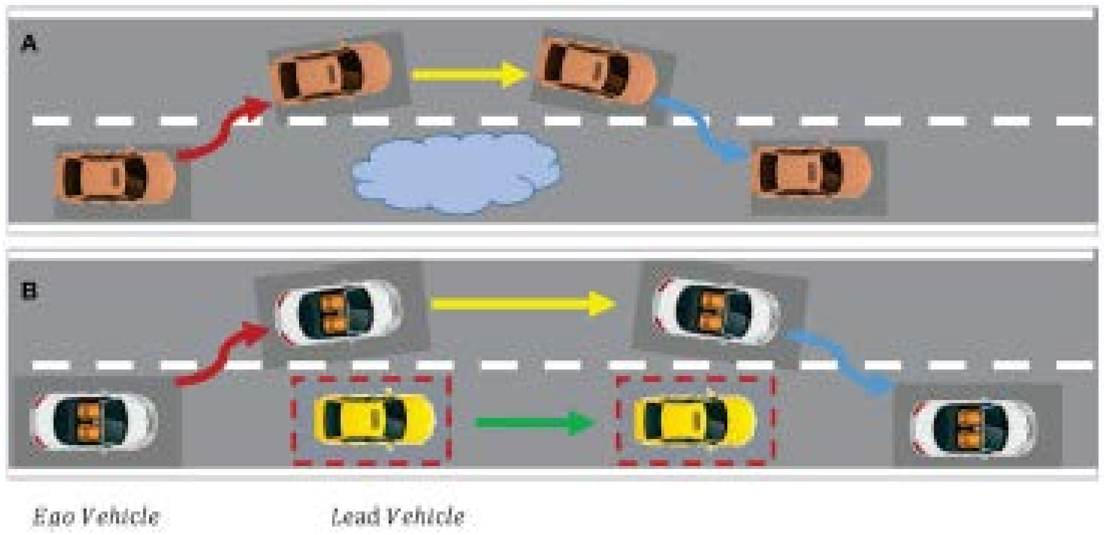



If change has happened (here slight deviation), can re-plan and adjust

My Initial take on the problem

Planning for Safe Abortable Overtaking Maneuvers

- Lane keep/change while ensuring sufficient clearance for safety
- Abort and merge back if safety is compromised
- Investigate performance



Typical scenario of overtaking. (A) Overtaking a static object; (B) Overtaking a dynamic object

- Initial goal was to leverage the advantages of low-level controller like MPC(real-time response, flexibility, more safety) for dynamic traffic maneuvers instead of using typical high-level planner.

- **But this proved to be quite challenging!!**

Gaussian Process based model predictive control for overtaking in autonomous driving, 2021

Research Pathway Revision

Adopted a new research direction to align with project objectives.

RWTH AACHEN UNIVERSITY

Vehicle Guidance in AD Software Stack

•**Objective**: Comfortable, efficient, and safe navigation.

•**Key Task**:
•Ensure vehicle safety, e.g., avoid collisions
•Identify and implement enhancements in the MPC-planner to better plan trajectories around dynamic objects.

•**Trajectory Planning**: : ACDC MPC-planner's *j_dyn* cost term.

•**Example Drawbacks**:

- Current position focus
- No vehicle geometries
- Single object consideration

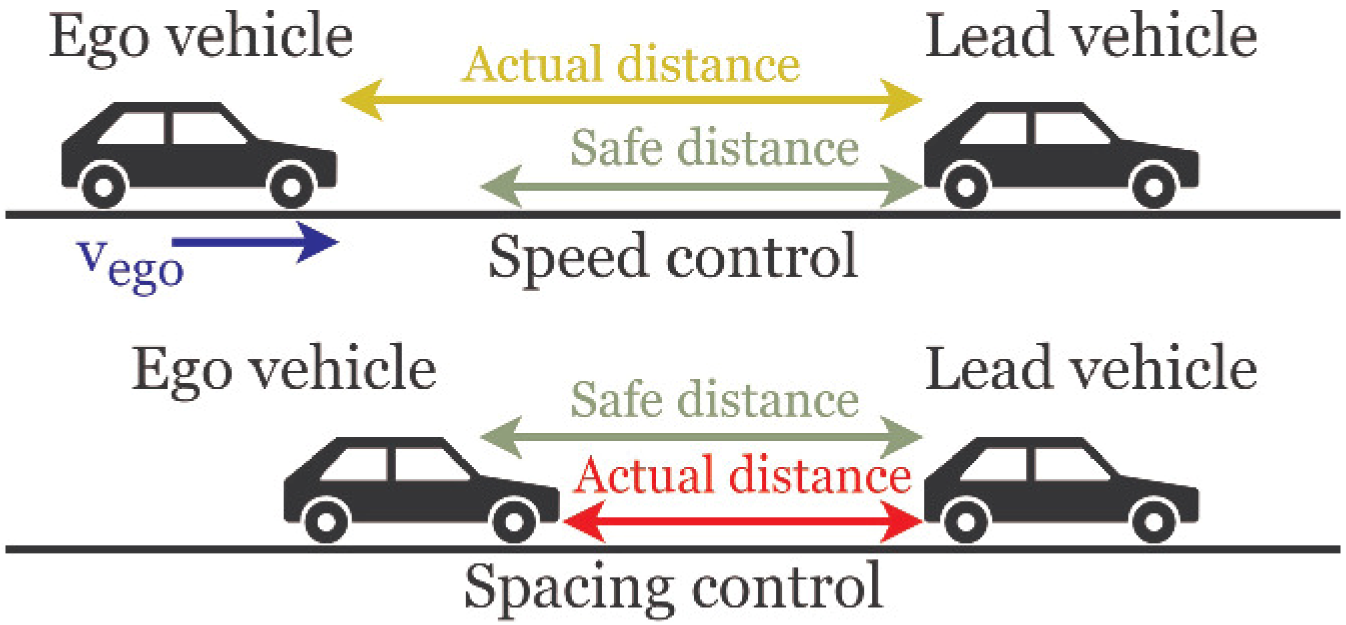


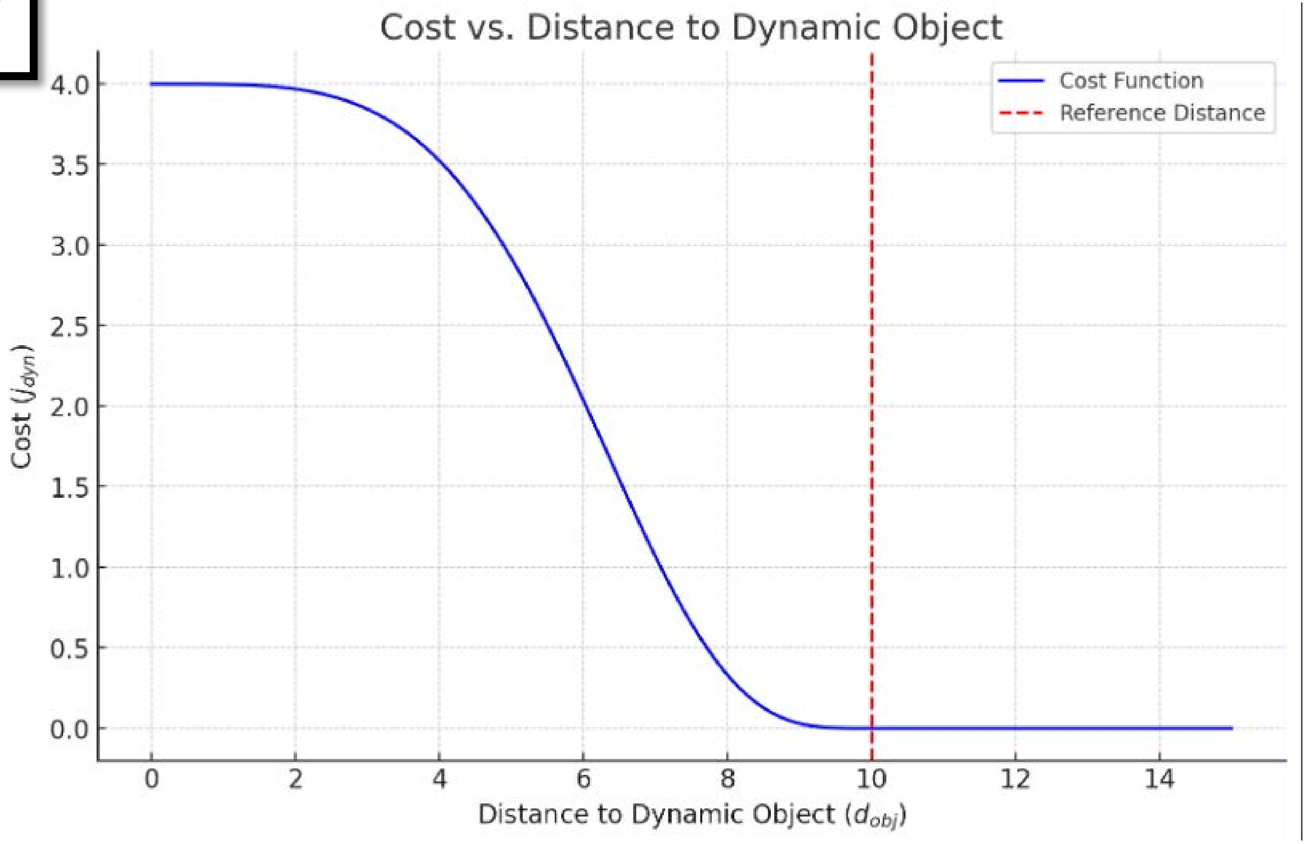
Fig from: Machine Learning-Based Fault Injection for Hazard Analysis and Risk Assessment

Cosine-based Cost Function

$$j_{dyn} = \left(w_{dyn} \cdot \left(\cos\left(\frac{\pi \cdot {d_{obj}}^2}{{d_{ref,obj}}^2}\right) + 1\right)\right)^2 \qquad \text{if} \quad d_{obj} < d_{ref}$$

Using a cosine-based cost function in the context of vehicle guidance and trajectory planning offers several advantages:

1. **Non-linearity**
2. **Smoothness**
3. **Boundaries**:
4. **Flexibility**
5. **Intuitive Thresholding**
6. **Computational Efficiency**
7. **Generalization**
8. **Safety Emphasis**



While there are several other functions that can be explored, the cosine-based cost function serves as a decent starting point for a more robust, efficient, and comprehensive trajectory planning system.

Constant Velocity and Acceleration Model

Previous Method: Distance computed using current dynamic object position and predicted positions of ego vehicle.

$$dynObjDist = \sqrt{(dynObjX - x[0])^2 + (dynObjY - x[1])^2}$$

Improved Distance Calculation in MPC : Distance computation considers future positions of the dynamic object, factoring in its current velocity and acceleration over time *t* for more proactive/accurate trajectory planning and enhanced safety.

$$dynObjDist = \sqrt{\left(dynObjX + \left(dynObjVx * t + \frac{1}{2} * dynObjAx * t^2\right) - x[0]\right)^2 + \left(dynObjY + \left(dynObjVy * t + \frac{1}{2} * dynObjAy * t^2\right) - x[1]\right)^2}$$

Incorporating Vehicle Geometry

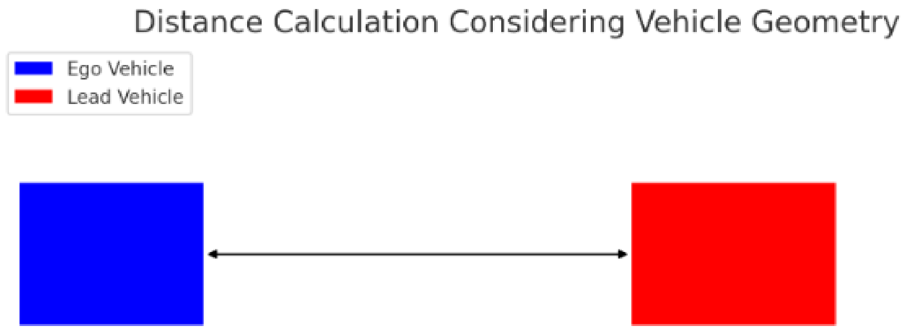
1.Geometry Consideration: Instead of treating vehicles as point objects, the modified equation accounts for the actual sizes of the vehicles, making distance calculations more accurate.

2.Conservative Estimation: Using the lengths of the vehicles ensures a conservative distance estimation, further emphasizing safety.

The final equation for dynamic Object Distance which accounts for vehicle geometry is:

$$\boxed{\text{DynObjDist} = \text{Eucledian Distance} - \text{Geometric adjustments}}$$

This represents half of the combined lengths of the two vehicles.



Stopping Distance for aggressive Ego Vehicles

Idea: Eliminating tunable parameters like *d_ref,obj* with non-tunable or inherent measures can help make the cost function more robust and reduce the need for extensive calibration.

Method:

$$\mathrm{stoppingDist}(v, a_{\max}) = \frac{v^2}{2 \times a_{\max}}$$

Here, a_{max} is the maximum longitudinal deceleration.
In our case, it was found to be -5.0 m/s^2.

Replace *d_ref,obj* with the vehicle's stopping distance, calculated based on its current speed and braking capabilities.

- Inherently adapts to the vehicle's dynamics.
- Naturally represents a safety margin as it ensures that the vehicle can come to a stop before colliding.

Problem: Variability of Stopping Distance:
The stopping distance of a vehicle is a function of its current speed and the braking capabilities/conditions. Inserting such a dynamic variable into the cosine function can cause the cost function's shape to change constantly and can lead to abrupt changes in the cost landscape, making optimization challenging.

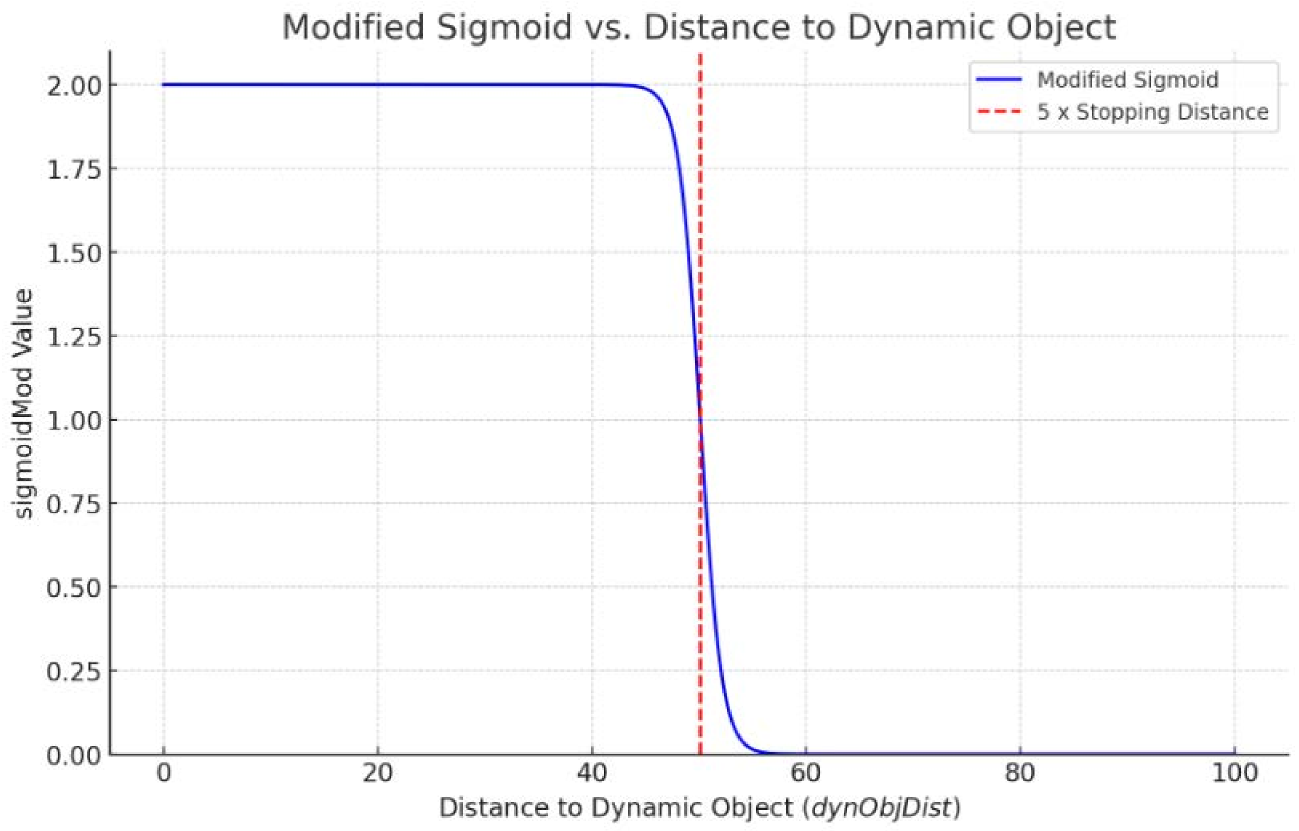
Need for a new term

Need: Introduce a clear, adaptive threshold where the system's behavior transitions from emphasizing efficiency to emphasizing safety.
Choice: Sigmoid(Smooth Transition, Bounded)

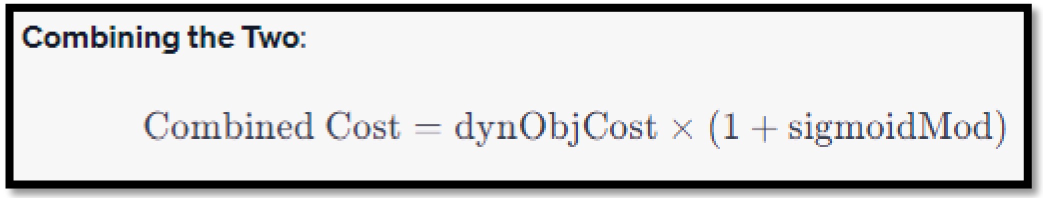
$$\boxed{\text{sigmoidMod} = \frac{2}{1 + \exp(k \times (\text{dynObjDist} - 5 \times \text{stoppingDist}))}}$$

1.Enhanced Behavior in Close Proximity: When the vehicle is very close to the dynamic, the sigmoid function approaches its upper limit, which, when multiplied with the original cost, emphasizes the importance of not getting any closer.

2.Safety Margin: The inflection point at $5 \times$ StoppingDistance introduces an inherent safety margin.



Implications of Combined Cost


Combining the Two:

$$\text{Combined Cost} = \text{dynObjCost} \times (1 + \text{sigmoidMod})$$

•**Dynamic Adaptability:** Adaptive to the vehicle's current speed and maximum braking deacceleration limit. This introduces a level of adaptability to the system, making it more responsive to real-time scenarios.

•**Retention of Original Behavior:** By multiplying the original cost with (1 + sigmoid), the base behavior of the trajectory planner is preserved as it doesn't introduce multiplicities or divide the cost function, which could have led to potential inconsistencies or discontinuities

•**Adaptive Safety Emphasis:** As the sigmoid increases, the combined cost can go up to 3 times the original cost. This happens when the ego vehicle is very close to the dynamic object within its stopping distance, thus "penalized" more heavily in these situations, emphasizing safety.

Handling Multiple Objects

Aim: Improve the cost function to account for multiple dynamic objects in the environment, prioritizing based on the ***Time to Collision*** (TTC = Relative Distance/ Relative speed), giving importance to objects that pose a more immediate risk.

1.Dynamic Weighting using Inverse TTC:

1. Instead of a static weighting factor w_dyn, each object is assigned a dynamic priority based on its normalized inverse TTC (1/TTC).
2. This ensures that objects that are more imminent in terms of collision risk are given a higher priority in the cost computation.

$$\text{Inverse TTC}_i = \begin{cases} \frac{1}{\text{TTC}_i} & \text{if } \text{TTC}_i \neq 0 \\ 1e6 & \text{otherwise} \end{cases}$$

$$w_{\text{dyn, i}} = \frac{\text{Inverse TTC}_i}{\sum_{j=1}^{N} \text{Inverse TTC}_j}$$

2. Summation of Costs:

1. For each object, compute its individual cost using the updated w_dyn (based on its priority).
2. Sum up the costs from all objects to get the total cost.

$$J_{\text{total}} = \sum_{i=1}^{N} j_{\text{dyn, i}}(w_{\text{dyn, i}})$$

RWTHAACHEN UNIVERSITY

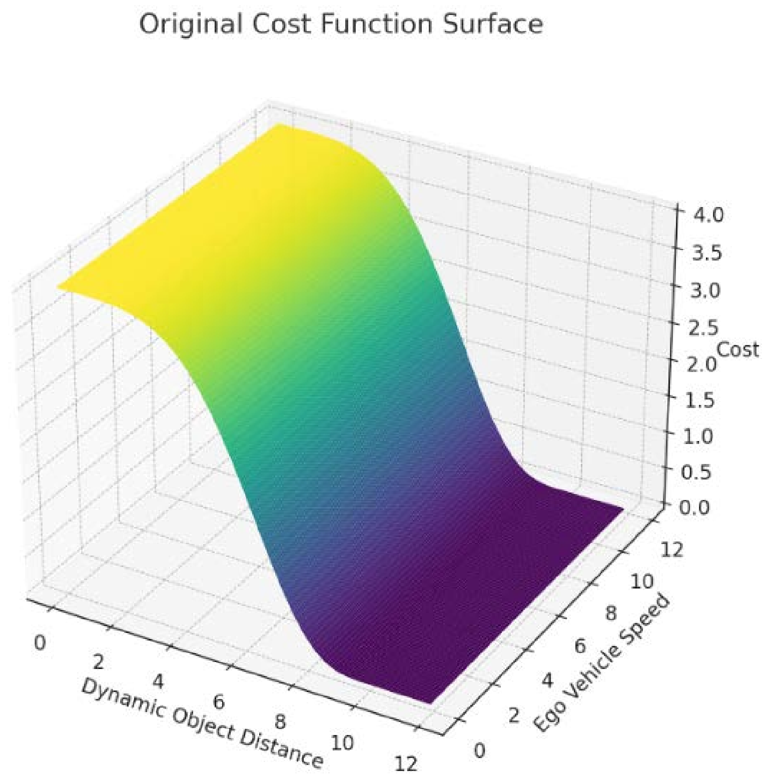
Final Term

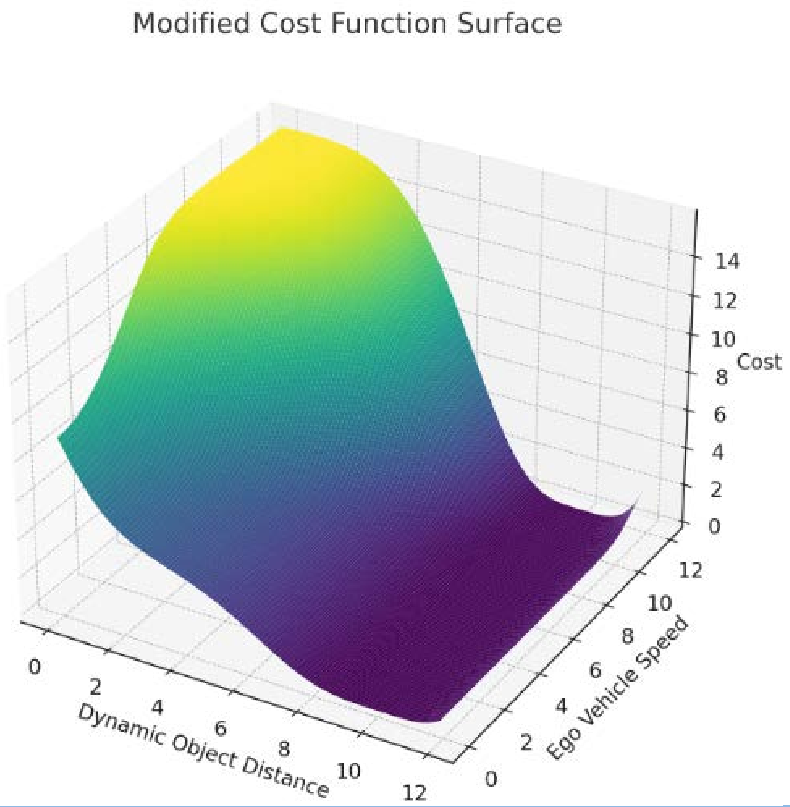
Final Cost term:

$$\left(\left(\cos\left(\pi \times \frac{d_{\text{modified}}^2}{d_{\text{dynObjRef}}}\right)^2 + 1\right) \times (1 + \text{sigmoid}(d_{\text{modified}}, k, d_{\text{stopping}})) \times w_{\text{dyn}}\right)^2$$

Dynamic Weighting Factor:

$$w_{\text{dyn}} = \max_i \left(\frac{\text{Inverse TTC}_i}{\sum_{j=1}^{N} \text{Inverse TTC}_j}\right)$$



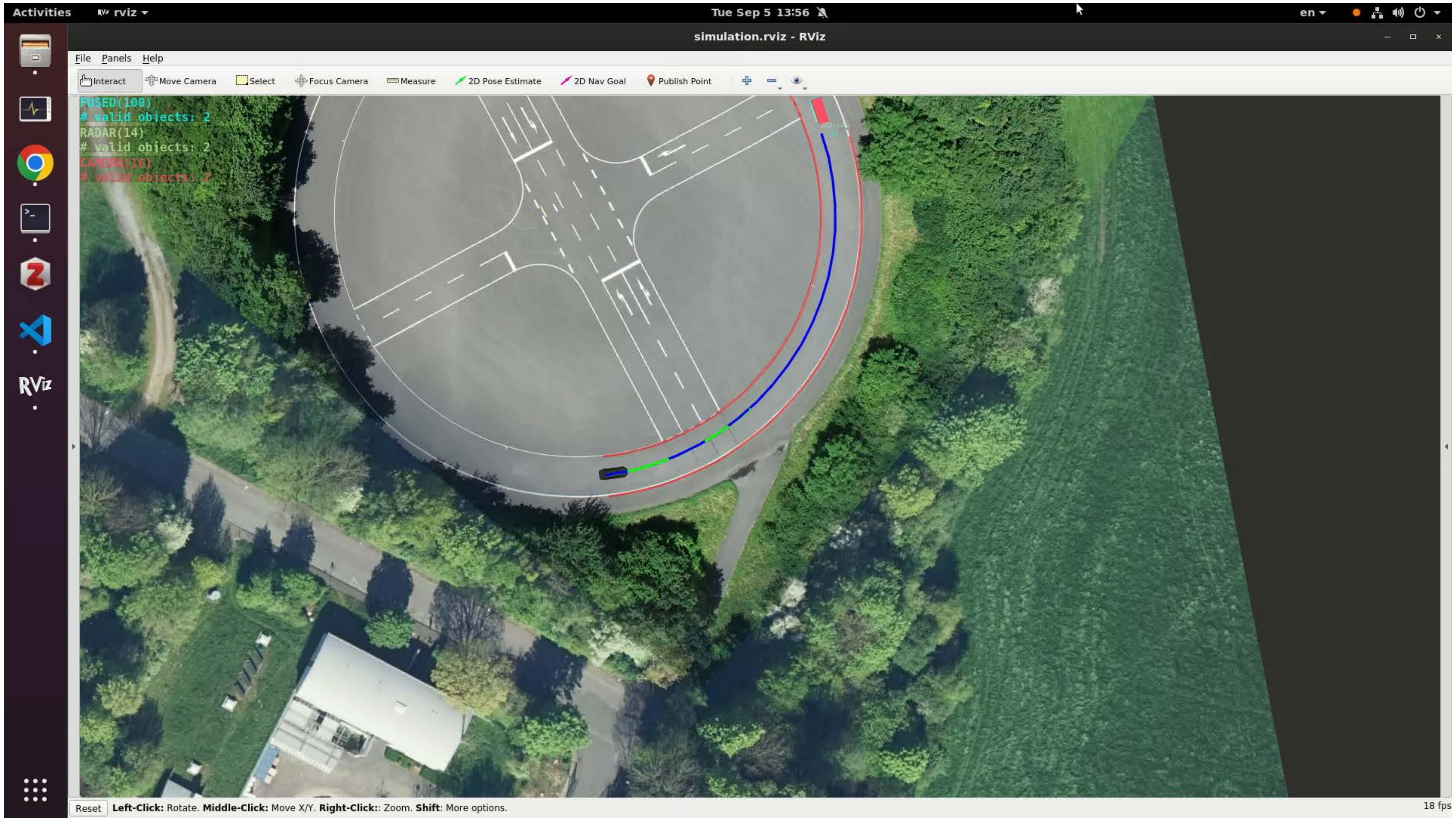


Choosing one Highest Priority Target:

•We opted to focus on the single highest-priority object as our critical target.

•This decision was from a computational and validation standpoint, adapting the code to return priorities for every object would also require significant refactoring with limited capability to validate multi-object considerations in our current object dataset.

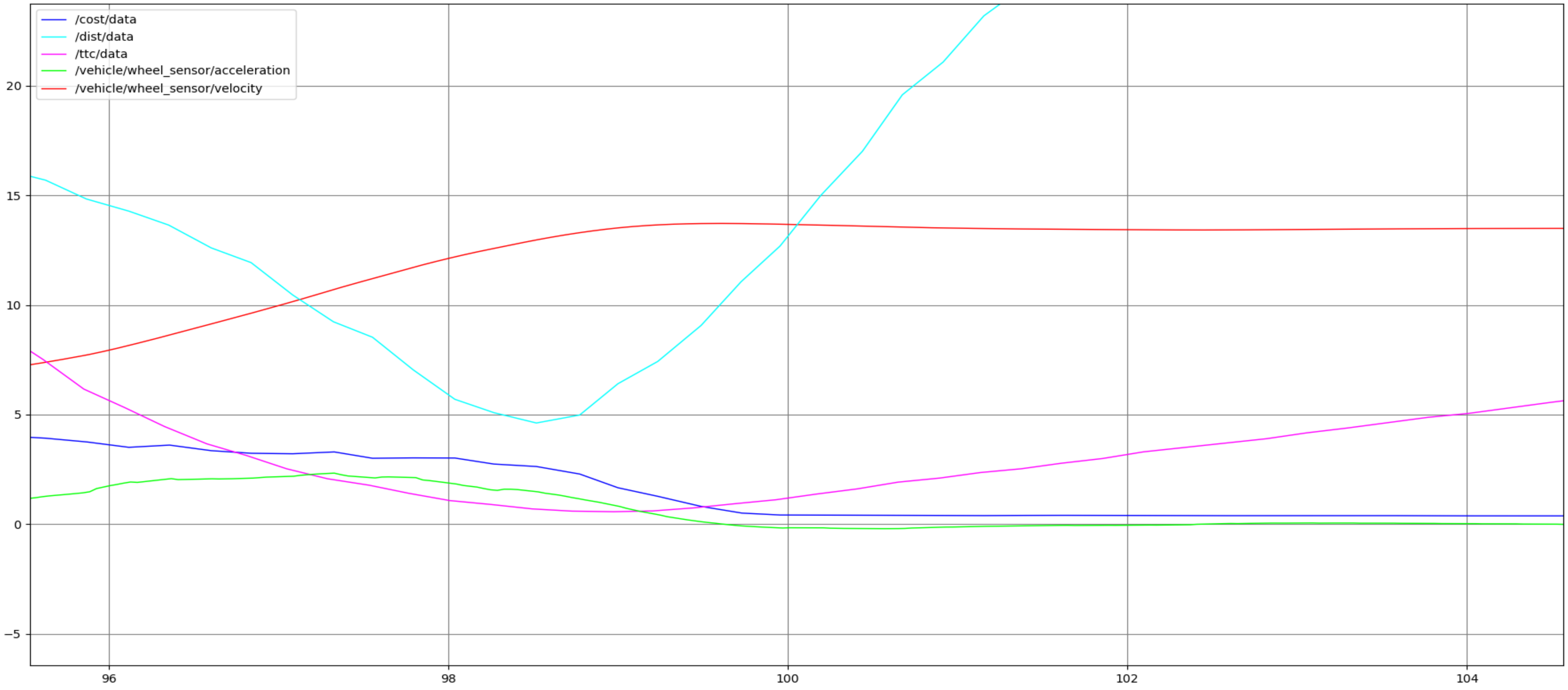
RViz



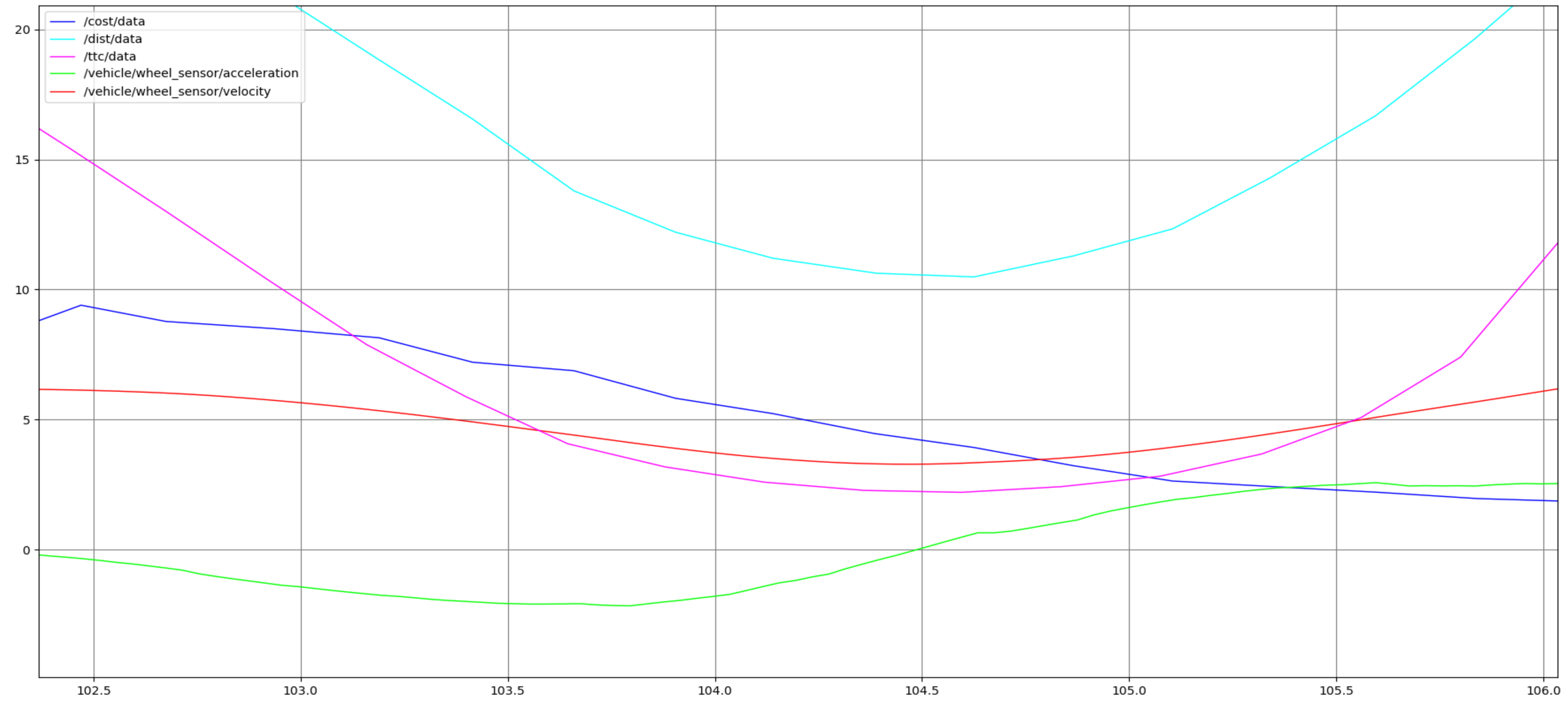
```
FUSED(100)
# valid objects: 2
RADAR(14)
# valid objects: 2
CAMERA(16)
# valid objects: 2
```

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Zoom. **Shift**: More options.

Performance Analysis – Original Cost Function



Performance Analysis – Modified Cost Function



Summary

Safety-centric decision-making at every step

Understand: ACDC Course’s Vehicle Guidance Software Stack and the study the problem in hand.

Improving Base Functionality: Recalibrated the value of Dynamic Object Distance, accounting for its predicted trajectory and the geometry of both vehicles.

Extending with Sigmoid Modulation: An added safety condition based on stopping distance.

Dynamic Weighting Factor :Assign higher priority to vehicle based on Inverse TTC, rather than closest leading vehicle.

Performance Analysis: Evaluate the system's decisions, both from a visual standpoint in Rviz simulation and by analyzing the rosbag data in rqt_plot with performance metrics like TTC, ego velocity/acceleration fluctuations, and cost function variations.

Coming back to Initial problem…

Key References:

Kabzan, J., Hewing, L., Liniger, A. and Zeilinger, M.N.
Learning-based model predictive control for autonomous racing.
IEEE Robotics and Automation Letters, 2019, 4(4), pp.3363-3370.

Kabzan, J., et. al.
AMZ Driverless: The Full Autonomous Racing System.
arXiv preprint arXiv:1905.05150, 2019.

B. Alrifaee. Networked Model Predictive Control for Vehicle Collision Avoidance. PhD thesis, RWTH Aachen University, 2017.

Problem Statement

Aim: Enforce a minimum safety distance between vehicles on "hard" constraint level (no violation) rather than providing a “soft” penalty in cost function.

The collision avoidance constraint can be

$$\|\mathbf{y}^{(i)}(t+k)-\mathbf{y}^{(j)}(t+k)\| \geq d_{safe}^{(i,j)},\ k=1,\ldots,H_p,$$

where $\mathbf{y}^{(i)}(t+k)$ and $\mathbf{y}^{(j)}(t+k)$ denote the predicted positions of vehicles v_i and v_j, respectively, at time $t+k$ obtained at time t, $d_{safe}^{(i,j)}$ specifies the distance required between the positions of vehicles v_i and v_j that guarantees the collision avoidance, and H_p is the prediction horizon. This formulation corresponds to a circular area with a radius of $d_{safe}^{(i,j)}$.

The constraint function can be defined as follows:

$$c_c^{(i,j)}(\mathbf{y}^{(i)}(t+k),\mathbf{y}^{(j)}(t+k)) = d_{safe}^{(i,j)} - \|\mathbf{y}^{(i)}(t+k)-\mathbf{y}^{(j)}(t+k)\|.$$

Building the platform

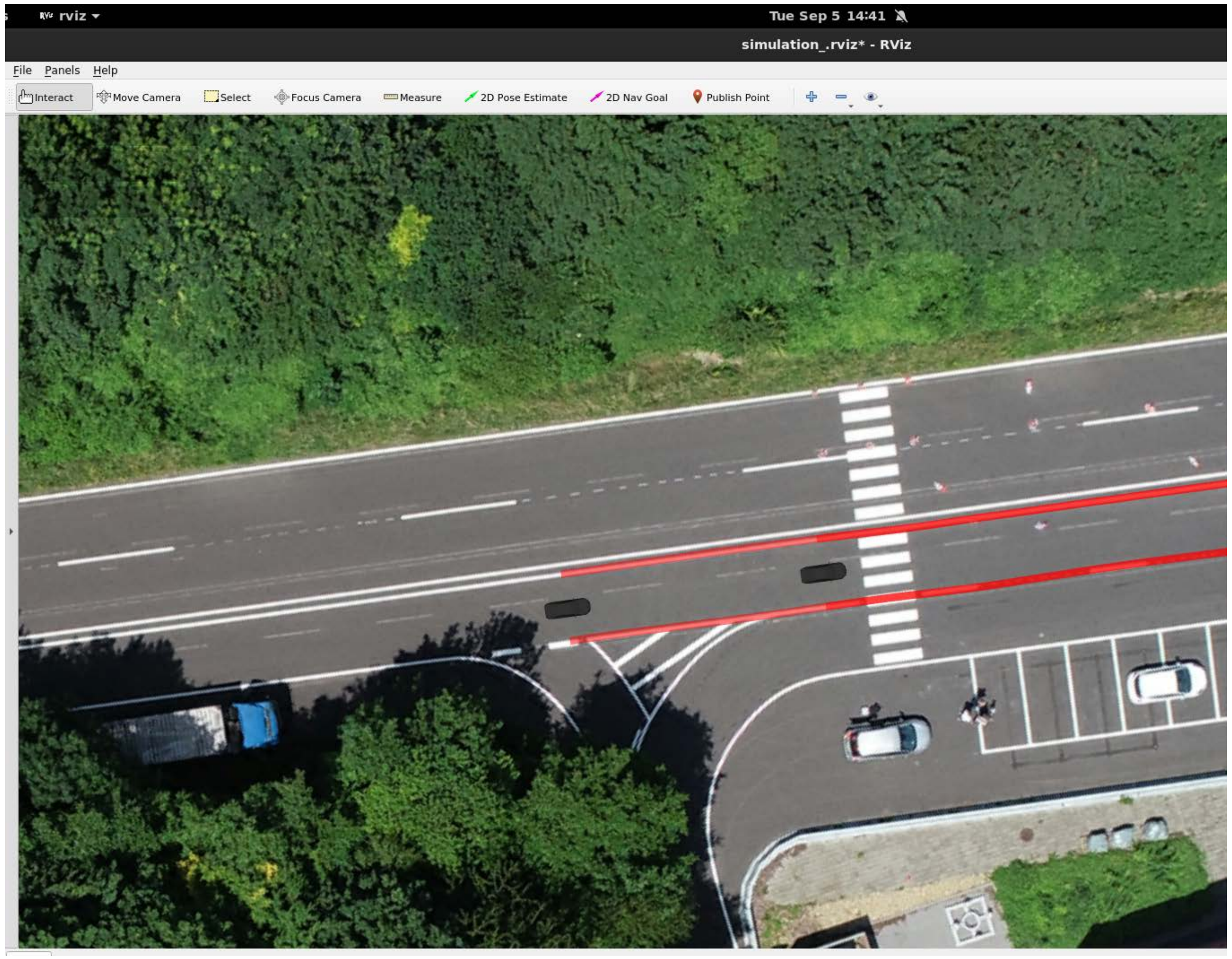
Modified Launch file to create two nodes for each vehicle (consistent with flatland state publisher):

Cloned and modified urdf and .yaml files appropriately to publish state information in separate topics.

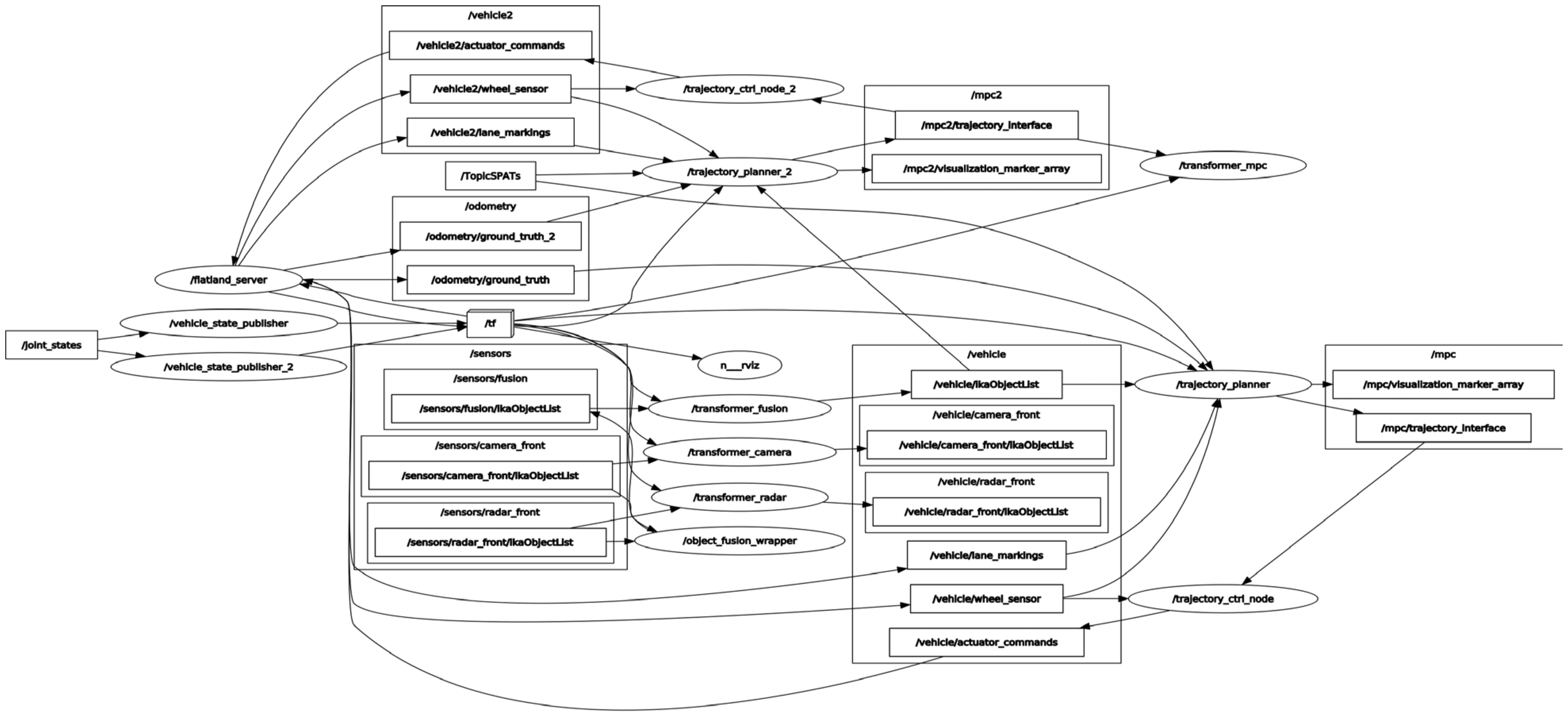
To Accommodate Overtaking Maneuvers:

Scale the vehicles to 0.5 and adjust the wheel joint positions accordingly in URDF file.

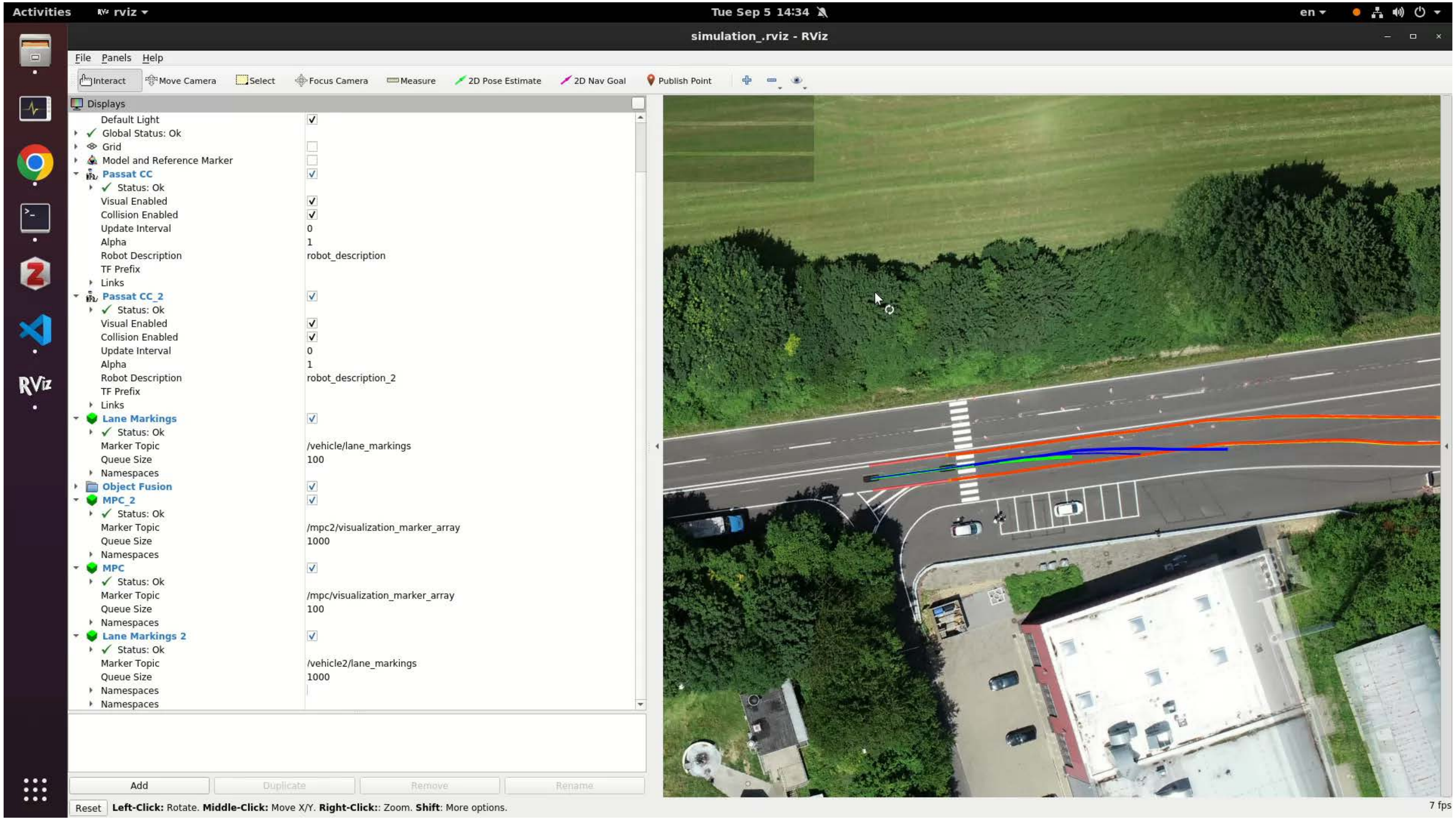
Set the default Reference Points from half(midpoint) to quarter of the sum of left and right boundary.



rqt_graph


/vehicle2
/vehicle2/actuator_commands
/vehicle2/wheel_sensor
/vehicle2/lane_markings
/trajectory_ctrl_node_2
/mpc2
/mpc2/trajectory_interface
/mpc2/visualization_marker_array
/transformer_mpc
/TopicSPATs
/trajectory_planner_2
/odometry
/odometry/ground_truth_2
/odometry/ground_truth
/flatland_server
/vehicle_state_publisher
/joint_states
/vehicle_state_publisher_2
/tf
n__rviz
/sensors
/sensors/fusion
/sensors/fusion/ikaObjectList
/sensors/camera_front
/sensors/camera_front/ikaObjectList
/sensors/radar_front
/sensors/radar_front/ikaObjectList
/transformer_fusion
/transformer_camera
/transformer_radar
/object_fusion_wrapper
/vehicle
/vehicle/ikaObjectList
/vehicle/camera_front
/vehicle/camera_front/ikaObjectList
/vehicle/radar_front
/vehicle/radar_front/ikaObjectList
/vehicle/lane_markings
/vehicle/wheel_sensor
/vehicle/actuator_commands
/trajectory_planner
/mpc
/mpc/visualization_marker_array
/mpc/trajectory_interface
/trajectory_ctrl_node

Scaled carS in simulation



Default Light	✓
Global Status: Ok	
Grid	
Model and Reference Marker	
Passat CC	✓
Status: Ok	
Visual Enabled	✓
Collision Enabled	✓
Update Interval	0
Alpha	1
Robot Description	robot_description
TF Prefix	
Links	
Passat CC_2	✓
Status: Ok	
Visual Enabled	✓
Collision Enabled	✓
Update Interval	0
Alpha	1
Robot Description	robot_description_2
TF Prefix	
Links	
Lane Markings	✓
Status: Ok	
Marker Topic	/vehicle/lane_markings
Queue Size	100
Namespaces	
Object Fusion	✓
MPC_2	✓
Status: Ok	
Marker Topic	/mpc2/visualization_marker_array
Queue Size	1000
Namespaces	
MPC	✓
Status: Ok	
Marker Topic	/mpc/visualization_marker_array
Queue Size	100
Namespaces	
Lane Markings 2	✓
Status: Ok	
Marker Topic	/vehicle2/lane_markings
Queue Size	1000
Namespaces	
Namespaces	

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Zoom. **Shift**: More options.

Combined Topic and Frame Transform

Custom Message Type: A new message type to collate and publish a single combined message from several distinct message sources of Ego Pose Data(Odometry), Vehicle wheel sensor data and MPC prediction data.

Processing Vehicle Trajectory: transform the trajectory data of one vehicle (vehicle_body_2) from its own local reference frame (laser_front_ref_2) to another vehicle's reference frame (laser_front_ref_1).

- If successful, the point is transformed, and stored in global variables for constraint processing.
- If the transformation fails, a warning is issued, and the function returns early(SEGMENTATION ERROR CONCERN).

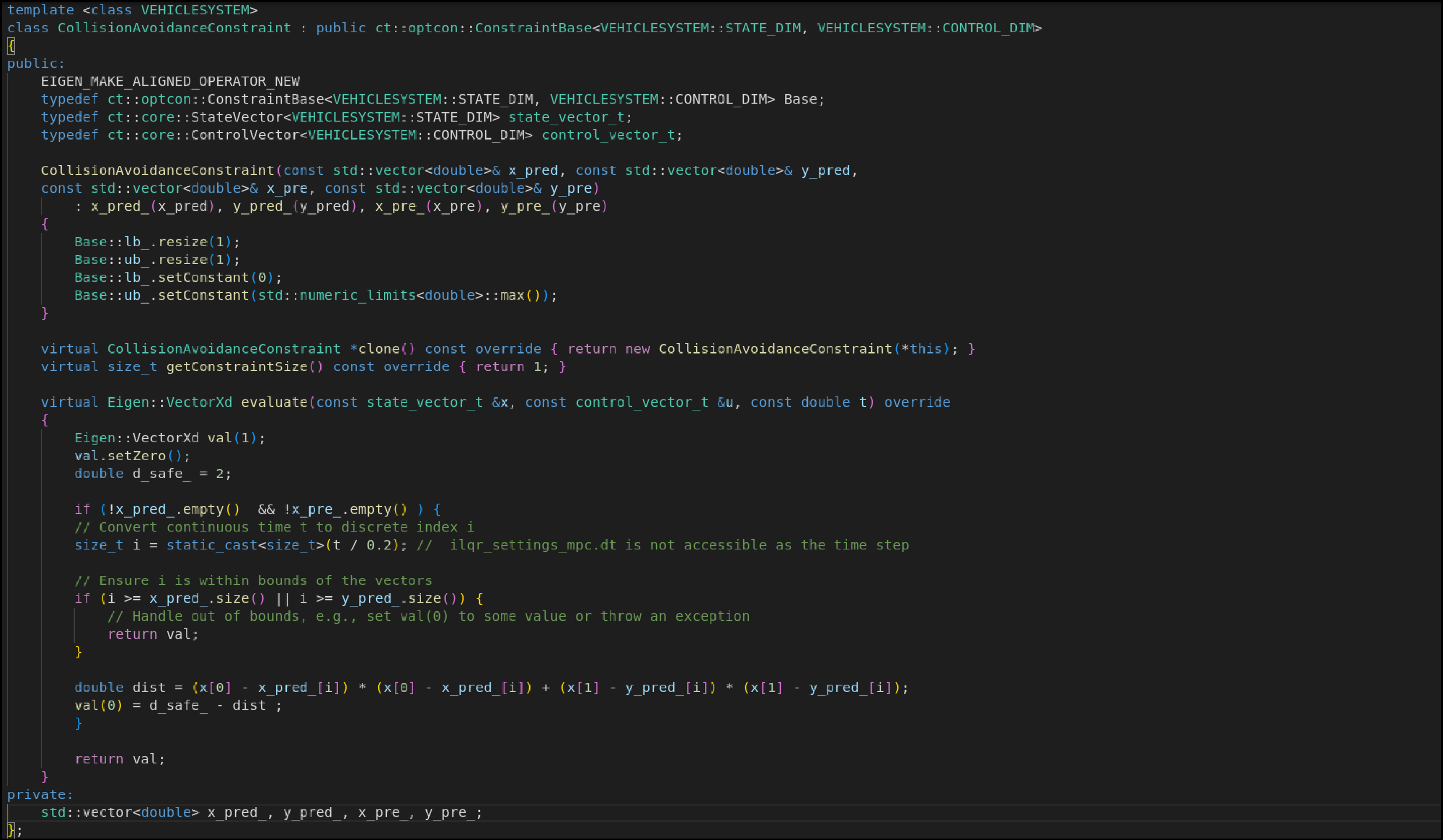
```
[trajectory_planner-7] process has died [pid 115858, exit code -6, cmd
/home/rosuser/ws/catkin_workspace/devel/lib/trajectory_planner/trajectory_p
lanner __name:=trajectory_planner __log:=/home/rosuser/.ros/log/08266176-
46c8-11ee-9527-04d9f51ce74d/trajectory_planner-7.log]. log file:
/home/rosuser/.ros/log/08266176-46c8-11ee-9527-
04d9f51ce74d/trajectory_planner-7*.log
```

Failed attempt on fixing reference lines…

Objective: Generate accurate and smooth and reference path for a vehicle, based on left and right boundary data.

•**Higher Resolution**: Increase number of points in the reference path to capture more details of the boundaries.
•**Higher Order Spline Interpolation**: Convert boundary points into cubic spline representations using Eigen's Spline Interpolation method.
•**Weighted Sampling:** Prioritize immediate points for sampling.
•**Projection**: Project the boundary points onto the closest spline to correct any deviations.
• **Iterative adaptive resampling** (**for smooth and reliable** path)**:**

  •Interpolate → Resample path –> Adjust Interpolation parameters based on distance between resampled points → Iterate.

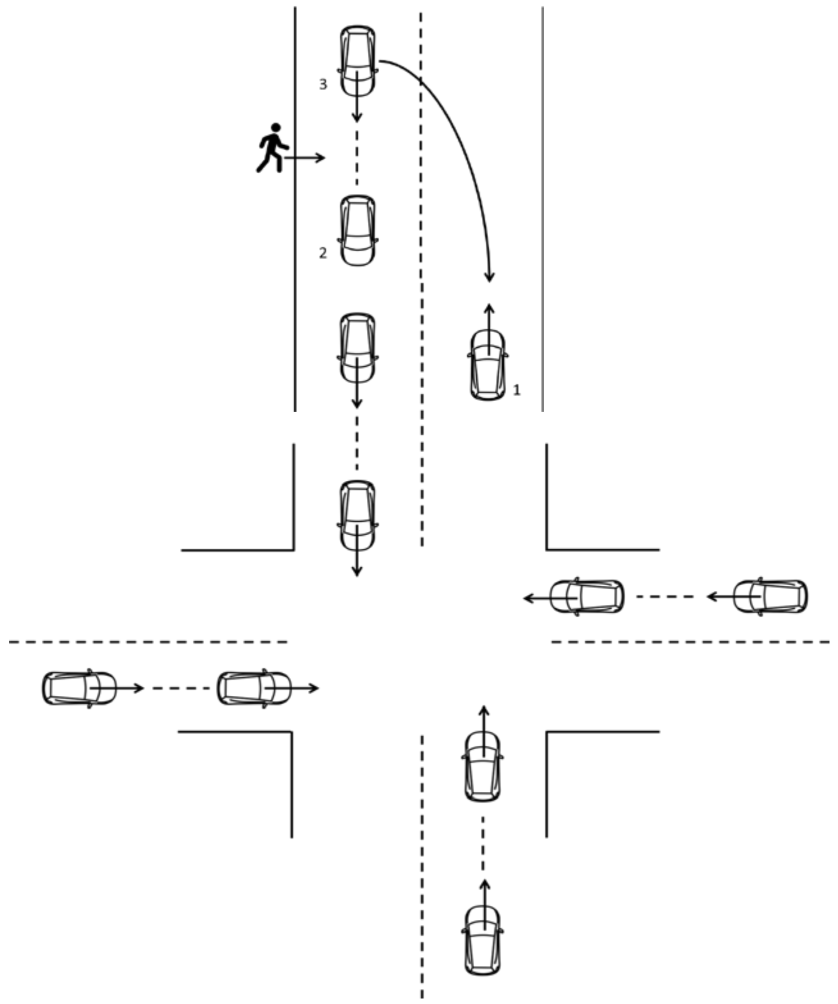

```cpp
template <class VEHICLESYSTEM>
class CollisionAvoidanceConstraint : public ct::optcon::ConstraintBase<VEHICLESYSTEM::STATE_DIM, VEHICLESYSTEM::CONTROL_DIM>
{
public:
    EIGEN_MAKE_ALIGNED_OPERATOR_NEW
    typedef ct::optcon::ConstraintBase<VEHICLESYSTEM::STATE_DIM, VEHICLESYSTEM::CONTROL_DIM> Base;
    typedef ct::core::StateVector<VEHICLESYSTEM::STATE_DIM> state_vector_t;
    typedef ct::core::ControlVector<VEHICLESYSTEM::CONTROL_DIM> control_vector_t;

    CollisionAvoidanceConstraint(const std::vector<double>& x_pred, const std::vector<double>& y_pred,
    const std::vector<double>& x_pre, const std::vector<double>& y_pre)
        : x_pred_(x_pred), y_pred_(y_pred), x_pre_(x_pre), y_pre_(y_pre)
    {
        Base::lb_.resize(1);
        Base::ub_.resize(1);
        Base::lb_.setConstant(0);
        Base::ub_.setConstant(std::numeric_limits<double>::max());
    }

    virtual CollisionAvoidanceConstraint *clone() const override { return new CollisionAvoidanceConstraint(*this); }
    virtual size_t getConstraintSize() const override { return 1; }

    virtual Eigen::VectorXd evaluate(const state_vector_t &x, const control_vector_t &u, const double t) override
    {
        Eigen::VectorXd val(1);
        val.setZero();
        double d_safe_ = 2;

        if (!x_pred_.empty()  && !x_pre_.empty() ) {
        // Convert continuous time t to discrete index i
        size_t i = static_cast<size_t>(t / 0.2); //  ilqr_settings_mpc.dt is not accessible as the time step

        // Ensure i is within bounds of the vectors
        if (i >= x_pred_.size() || i >= y_pred_.size()) {
            // Handle out of bounds, e.g., set val(0) to some value or throw an exception
            return val;
        }

        double dist = (x[0] - x_pred_[i]) * (x[0] - x_pred_[i]) + (x[1] - y_pred_[i]) * (x[1] - y_pred_[i]);
        val(0) = d_safe_ - dist ;
        }

        return val;
    }
private:
    std::vector<double> x_pred_, y_pred_, x_pre_, y_pre_;
};
```

Research Potential…

Decentralized Net-MPC: Dealing with the action of other agents with respect to their future intention, while making decisions of the agent's own control actions.

Smart CAM: Enhance CAM (Cooperative Awareness Message) with more informed information for coordinated planning.

Behavior Conditioning: Based on just the predictions, Predict and understand agent behavior patterns. If discrepancies or anomalies are found, condition the behavior to align with desired outcomes.



B. Alrifaee. Networked Model Predictive Control for Vehicle Collision Avoidance

Research Lessons

- It's essential to remain adaptable. Research is often non-linear and doesn't always proceed as initially planned, and new findings or challenges can necessitate a shift in direction. Embracing such changes can lead to more meaningful outcomes.

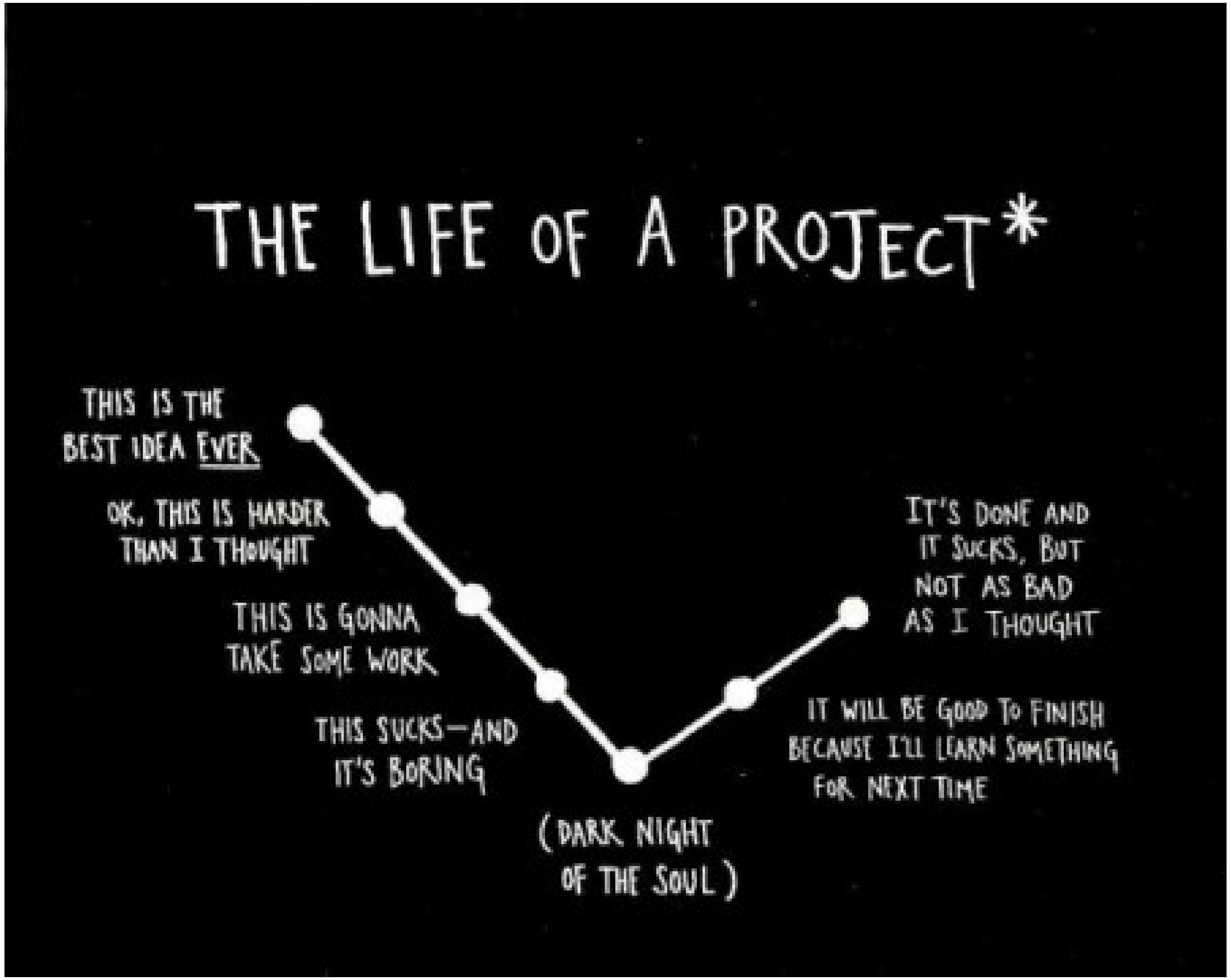
- Research is often iterative. An initial hypothesis or problem might evolve based on intermediate findings or external feedback.

- Working independently demands a high degree of self-motivation and initiative. Proactively seeking out resources, methodologies, and solutions becomes crucial., seeking external feedback can provide fresh perspectives, helping to identify overlooked issues or to validate findings.

- Documenting the reasons for changes and decisions made during the research process is crucial. This not only provides a clear record but also helps in analyzing the effectiveness of decisions.

- Essential to recognize and celebrate small achievements, especially when working alone.

Every new challenge is a new learning experience. Thank you!


THE LIFE OF A PROJECT*
THIS IS THE BEST IDEA EVER
OK, THIS IS HARDER THAN I THOUGHT
THIS IS GONNA TAKE SOME WORK
THIS SUCKS—AND IT'S BORING
(DARK NIGHT OF THE SOUL)
IT WILL BE GOOD TO FINISH BECAUSE I'LL LEARN SOMETHING FOR NEXT TIME
IT'S DONE AND IT SUCKS, BUT NOT AS BAD AS I THOUGHT

References

Liu, Wenjun, et al. "Gaussian process based model predictive control for overtaking in autonomous driving." *Frontiers in Neurorobotics* 15 (2021): 723049.

B. Alrifaee. Networked Model Predictive Control for Vehicle Collision Avoidance. PhD thesis, RWTH Aachen University, 2017.

Palatti, Jiyo, et al. "Planning for safe abortable overtaking maneuvers in autonomous driving." *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2021.

Kabzan, J., Hewing, L., Liniger, A. and Zeilinger, M.N.
Learning-based model predictive control for autonomous racing.
IEEE Robotics and Automation Letters, 2019, 4(4), pp.3363-3370.