Autonomous Control for Car-Trailer System

Aadit Patel
aaditp@seas.upenn.edu

Aadith Kumar
aadith@seas.upenn.edu

Orlando Lara Guzman
orlara@seas.upenn.edu

Sharon Richu Shaji
sshaji@seas.upenn.edu

***Abstract*—In this project, we designed and built an autonomous single-trailer car that can safely and efficiently navigate through an environment with obstacles. To achieve this objective, we designed a control system that integrates a non-linear trajectory optimization framework and a Pure Pursuit controller to accurately track the desired trajectory of the car-trailer system, taking into account the non-linear dynamics of the model. We conducted experiments using obstacles such as cans and developed a controller that enabled the car-trailer system to avoid some of the cans while directing the trailer to hit other specific ones. Our experiments involved different can orientations, and we present some qualitative results. Our findings demonstrate that the autonomous car-trailer system successfully hit selected cans and avoided others, while closely tracking the desired trailer trajectory. Here is a link to the GitHub Repo.**

***Index Terms*—Car-Trailer System, Model Predictive Control (MPC), Pure Pursuit**

I. Introduction

As self-driving cars become a reality, there will be an increasing demand for adding trailers and extensions to one's autonomous cars. However, controlling the motion of a car and trailer in a coordinated manner is a complex problem that requires the development of a good control system. In fact, current cars such as Tesla immediately disable autopilot on adding a trailer as their autopilot is not engineered for the car-trailer system.

The development of autonomous car-trailer systems can benefit the logistics and transportation industry by improving efficiency, reducing costs, and increasing safety. These systems could reduce the need for human drivers, making transportation more efficient and cost-effective while also reducing the risk of accidents caused by driver error.

Overall, solving this problem will help advance the development of autonomous car-trailer systems and contribute to the realization of safe and efficient self-driving cars that can tow trailers. By demonstrating the feasibility and performance of these systems, we can pave the way for real-world applications of autonomous car-trailer systems that can benefit society.

The control problem of a single-trailer car model can be challenging due to the complexity and non-linearity of the system dynamics. The behavior of the system is influenced by many factors, such as the weight and distribution of the load, the design of the trailer, and the interaction between the trailer and the towing vehicle.

The single-trailer car model is a highly coupled system, which means that the movement of the trailer is strongly influenced by the movement of the towing vehicle, and vice versa. This coupling can lead to complex interactions between the two parts of the system, making it difficult to predict or control the behavior of the trailer.

Furthermore, it is an under-actuated system. The system has multiple degrees of freedom (the velocity, position and orientation of the trailer and car) but only two control inputs (the steering of the towing vehicle, velocity).

The objective of the project is to design and build an autonomous single-trailer car that can safely and efficiently navigate through an environment with obstacles.

To achieve this objective, we designed a control system using a non-linear trajectory optimization framework and Pure Pursuit controller that can accurately track the car-trailer system trajectory accounting for the non-linear dynamics of the system.

To test the control system, we placed obstacles like cans in the environment and built a controller that ensures the car avoids the cans while the trailer is controlled to knock-off specific cans.

II. Related Work

Most of the research in this area comes from the autonomous trucking sector. Lots of commercial companies have made significant progress in modeling the combined system dynamics of cars and trailers, trajectory planning, and controls. [1] presents a dynamic model of a car with n trailers and proposes output feedback kinematic and feedback linearization kinetic controllers for trajectory tracking. This work performed experiments on their system dynamics model for a deferentially driven car with a trailer. [3] discusses the comparison between two controllers, the Linear Quadratic Regulator (LQR) and the Deep Deterministic Policy Gradient (DDPG), in the path following task of autonomously backing up a tractor-trailer with varying wheelbase, hitch length, and velocity. This work efficiently compares the performance of a learning based and a non-learning based controller for a car-trailer system.

The work done in [2] provides benchmarks for motion planning on roads to address the issue of repeatability in numerical experiments for motion planning of road vehicles. This work also provides references to a lot of common road vehicles along with their dynamics and the constraints in which they operate. The single-trailer car system is also described in this work.

[4] presents four real-time implementable nonlinear model predictive control (NMPC) formulations using the hitch angle

measurement for torque-vectoring control of an electric front-wheel drive car towing a trailer. [4] assumes the availability of measured or estimated hitch angle information. All the experiments performed in this work were done in simulation only.

Another type of control algorithm that is often seen in car-trailer systems is the sliding mode control (SMC) technique. In [5], they propose this type of controller in which slipping is modeled as unknown disturbances in the vehicle dynamics model. A model that takes into account the slippage of the wheels is more realistic and will lead to deriving a more robust controller. This work was also only tested in simulation.

[6] provides a solution for motion planning of a car-trailer system by using the system's flatness with the Cartesian coordinates of the trailer as the linearizing output. [6] using Frenet formulations for simplifying the calculations for handing angle constraints of the system.

III. Methodology

The methodology adopted in this work is depicted in figure (1). It consists of 4 sections. They are described as follows -

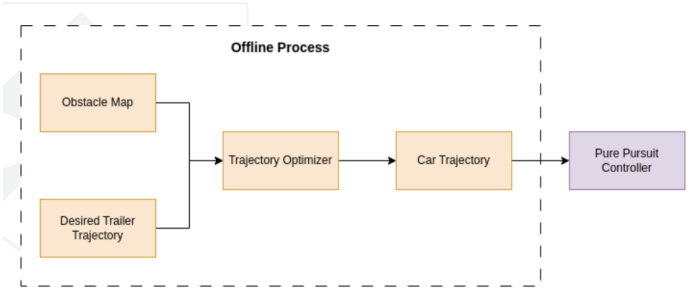

Fig. 1. Project Methodology Flowchart

A. Creating an Obstacle Map

An open area was mapped to allow for aggressive maneuvers for the car-trailer system. For our experiments, we choose the Wu and Chen upper lobby at Levine Hall, University of Pennsylvania. The map was created using the SLAM toolbox provided for ROS2. After map creation, obstacles were added using GIMP software on Ubuntu.

B. Creating the Desired Trailer Trajectory

In this work, we propose a control algorithm for the trailer to follow a given trajectory. Thus, the desired trailer trajectory is selected as waypoints on the map. We use RVIZ and ROS2 to log waypoints on our obstacle map. The trailer trajectory is designed to knock off cans from the environment, thus the trailer trajectory would go through some of the "obstacles" of the map and not go through the rest.

C. Trajectory Optimization for the Car

The low-level controller on the F1Tenth car can control the pose of the car. Thus, we need to map the trailer coordinates to their corresponding car coordinates. Given the desired trailer coordinates $(x_{trailer}, y_{trailer})$, we solve for the car coordinates (x_{car}, y_{car}), the car yaw with respect to the global frame (θ_{car}) and the hitch angle between the car and the trailer (β) that could get the trailer in the desired coordinates.

The car pose calculation is solved as an optimization problem. The objective of the optimization problem is to reduce the error between the desired trajectory of the trailer and the trailer trajectory computed using the optimization approach. By minimizing this objective, we get the car coordinates that would lead to the desired trailer trajectory.

D. Pure Pursuit Controller for the Car Trajectory

The car follows the computed car trajectory using a Pure Pursuit Controller. The Pure Pursuit controller takes the desired car waypoints as input and computes the required steering angle and velocity which is given as input to the car.

IV. Problem Formulation

A. Model

The car-trailer system model we have opted to use is derived from the single on-axle trailer model as discussed in [2].

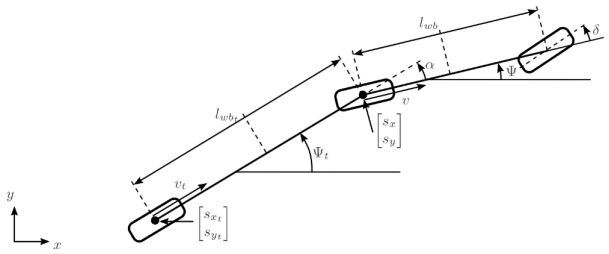

Fig. 2. Kinematic single-track model with one on-axle trailer

The states of the car trailer system can be expressed as :

$$\vec{x} = \begin{bmatrix} x_{car} \\ y_{car} \\ \theta_{car} \\ \beta \end{bmatrix} = \begin{bmatrix} {}^1x \\ {}^2x \\ {}^3x \\ {}^4x \end{bmatrix}, \vec{u} = \begin{bmatrix} v \\ \delta \end{bmatrix}$$

where the state $\vec{x}$ is the car position , yaw , and hitch angle β . The inputs $\vec{u}$ are the velocity of the car v and the steering angle δ.

The kinematic model an be seen in fig (2), and from that we can determine the dynamics equations to be as follows:

$$\begin{aligned} {}^1\dot{x} &= {}^1u \cos\left({}^3x\right) \\ {}^2\dot{x} &= {}^1u \sin\left({}^3x\right) \\ {}^3\dot{x} &= \frac{{}^1u}{l_{wb}} \tan\left({}^2u\right) \\ {}^4\dot{x} &= -{}^1u \left(\frac{\sin\left({}^4x\right)}{l_{wbt}} + \frac{\tan\left({}^2u\right)}{l_{wb}} \right) \\ f(\vec{x}) &= \dot{\vec{x}} \end{aligned}$$

In a discrete time dynamics setup, we can approximate this as:

$$\vec{x}_{k+1} = \vec{x}_k + f(\vec{x}) * dt$$

where dt is the time interval for which we discretize the dynamics.

B. MPC

We posed our trajectory optimization problem as a non-linear MPC problem to be solved across the trajectory offline with a pre-decided initial condition. The optimization problem was set up as follows:

1) Optimization variables: We wanted to obtain the trajectory of the car in space such that it achieves the desired trailer trajectory. We would like to solve for a sequence of states that allows us to do that. We solve for states on *knot points* on which we impose constraints and enforce our objective. The number of knot points N is a hyper-parameter that controls the trade-off between output accuracy and speed. More knot points mean more optimization variables and thus a larger optimization problem to solve.

For *N* knot points, we have *N* states and *(N-1) control vectors*:

$$X = \left[\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_i, \ldots, \vec{x}_N\right]$$
$$U = \left[\vec{u}_0, \vec{u}_1, \ldots, \vec{u}_i, \ldots, \vec{u}_{N-1}\right]$$

Further, we also keep the discretization time step of our continuous state dynamics as an optimization variable to allow the optimizer to use the knot points more efficiently. i.e., areas, where the state and the trajectory do not change can now be represented as a single knot point executed over a longer time step dt, and areas with a lot of dynamics and change to use more knot points to better represent that complex state evolution.

$$T = \left[\vec{dt}_0,\ \vec{dt}_1,\ \ldots,\ \vec{dt}_i\ \ldots,\ \vec{dt}_{N-1}\right]$$

2) Objective function: Our goal is to track the desired trailer coordinates of the cat trailer system. The state $\vec{x}$ of the system is sufficient to calculate the trailer coordinates. If $\vec{p}^{desired}$ is the desired trailer location, and we have *D* desired points, The transformation can be given as follows:

$${}^{world}\vec{p}_{trailer} = {}^{world}T_{car} \times {}^{car}\vec{p}_{trailer} = g(\vec{x})$$

$$g^{-1}({}^{world}\vec{p}_{trailer}, {}^{3}x, {}^{4}x) = \vec{x}$$

$$p_{trailer} = \begin{bmatrix} \cos({}^{3}x) & -\sin({}^{3}x) & {}^{1}x \\ \sin({}^{3}x) & \cos({}^{3}x) & {}^{2}x \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -l_{wb} - l_{wbt}\cos({}^{4}x) \\ -l_{wbt}\sin({}^{4}x) \\ 1 \end{bmatrix}$$

where ${}^{car}\vec{p}_{trailer}$ is the point of the trailer axle in the frame of the car. Thus our primary objective can be expressed as follows:

$$C_{track} = \sum_{i=0}^{N} K_{track}(\min_{k=0\ldots D}[g(\overrightarrow{x}_i) - \overrightarrow{p_K}^{desired}])^6 + K_{final}(g(x_n) - p_D)^2$$

Where the *K* values represent weights for each portion of the cost.The final cost is important as that is what makes the system want to reach the final goal.

Apart from that, we would like to get a smooth trajectory that minimizes general trailer movement unless required and keep the rate of change of velocity to be minimal to reduce rapid controls on the car. Thus we add additional cost to the objective penalizing sudden changes in velocity and actuation in steering. As we want the car to go fast we do not have a cost on speed, only acceleration.

$$C_{input} = \sum_{i=0}^{N-2} \left(K_{acc}({}^{0}u_i - {}^{0}u_{i+1}\right)^2 + K_{steer}{}^{0}u_i^2 \quad (1)$$

The final optimization objective becomes:

$$\min_{X,U,T} (C_{track} + C_{input})$$

3) Constraints: The constraints in optimization problem are listed below:

$$\forall \vec{x}_i \in X,\ \forall \vec{u}_i \in U,\ \forall dt \in T$$

$$\vec{x}_0 = g^{-1}(\vec{p}_0^{\,desired}, 0, 0) : \textit{Inital position constraint}$$

$$\vec{x}_i + f(\ \vec{x}_i\ ) * dt_i - \vec{x_{i+1}} = 0 : \textit{Dynamics constraint}$$

$$x_i \notin Obstacles\ : \textit{car cannot be in an obstacle}$$

$$\vec{u}_i \in [\vec{u}_{\max}, \vec{u}_{\min}]\ : \textit{Input saturation constraints}$$

$$\vec{x}_i \in [\vec{x}_{\max}, \vec{x}_{\min}] : \textit{state limits constraints}$$

4) warm start: Finally we also add a warm start which initializes the guess of the trajectory to be a line towards the goal. A better start would be to discretize the desired trajectory by N points and setting those points to be the guess locations

C. Pure Pursuit

In order to track a trajectory once given the car waypoints, a simple and robust controller is the Pure Pursuit [9]. This controller tracks the projection of a point at a 'look-ahead' distance from the robot and corrects the turning in proportion to the curvature to get to the point. It can be modified extensively for different applications. But as this is not the primary goal of the project, we have opted to go for the vanilla version with constant speed and look-ahead distance. We compute the curvature of the tracking point and scale it by a proportionality factor K_{gain} to command a steering angle delta as follows:

$$\delta = k_{gain} \frac{|y|}{L^2}$$

Where L is the lookahead distance and y is the lateral displacement of the target point in the robot frame. we command a fixed velocity to the controller, but we could use the offline computed Velocity profile as well.

V. Experiments

To test that we are correctly controlling the trailer of the car, we attempt to navigate through a set of obstacles. However, in our case, success isn't defined as just avoiding obstacles. A better measure of success is to first ensure that the car does not hit any obstacles at all and secondly that the trailer hits specific objective obstacles while avoiding all others. We will describe our setup and results below.

A. Hardware Setup

We use the F1Tenth car platform provided in the course as the driving car. It comes with Jeston NX compute onboard and is powered by a LiPO battery. We used a discarded car chassis of the same type as a trailer for the car. We fabricated a hitch by laser cutting acrylic boards and fastening them to a hitch shaft. Further, we attached a rotary encoder to the hitch that would allow us to read the hitch angle. We paired the encoder to an ESP32 to process the sensor input and then communicate it to the Jetson via serial communication.

Figure (3) shows the trailer mechanism as discussed in the above paragraph. Figure (4) shows the overall car-trailer system.

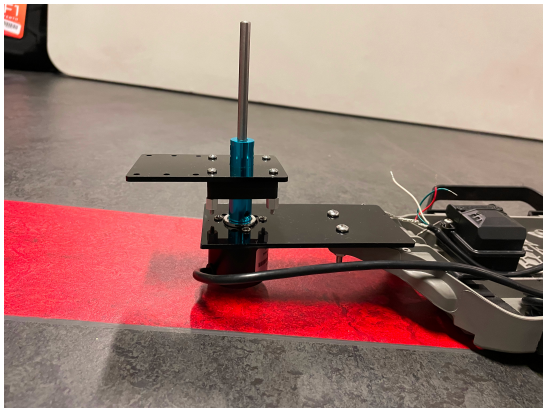

Fig. 3. Trailer setup in profile view

B. Simulation

As a simulation environment, we used the dynamics equations from the trajectory optimization to forward simulate the car's position, the trailer's position and the hitch position over time. The outputs can be seen in figures 5 and 7. Here we have plotted the mapped environment as black dots and you will notice some obstacles on the inside of the map representing cans. The proposed desired trajectory is shown in blue while the output trailer trajectory from our MPC is shown in red. For each time step, we have also plotted the corresponding car and hitch positions shown in cyan and magenta respectively.

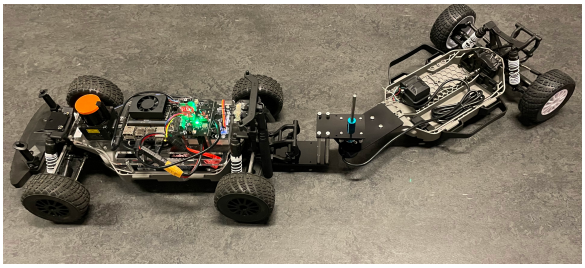

Fig. 4. Final Car and trailer setup in profile view

1) Experiment with 1 Can: For this experiment, we wanted to show that we were able to control the car-trailer system in such a way that the trailer could hit a soda can while still having the car avoid it. From figure 5, we can see the can in the top right portion of the map and the desired trailer trajectory is passing right through it. The car has its initial position in the top right-most part of the map and we desire that it go down and to the left. We can see that the car is able to successfully avoid it and you might notice that in this particular image, the trailer also avoids it. This was intentional since we were aware of the sim-2-real gap and made the assumption that given the error, the objective would be fulfilled. We were satisfied with this result because it seemed to be just a matter of tuning on the real car; we decided that it didn't make much sense to keep tuning in the simulation. For completeness, we made sure that the algorithm could accurately track the desired trajectory; we did this by **removing the constraint** to avoid obstacles and the result is shown in figure 6.

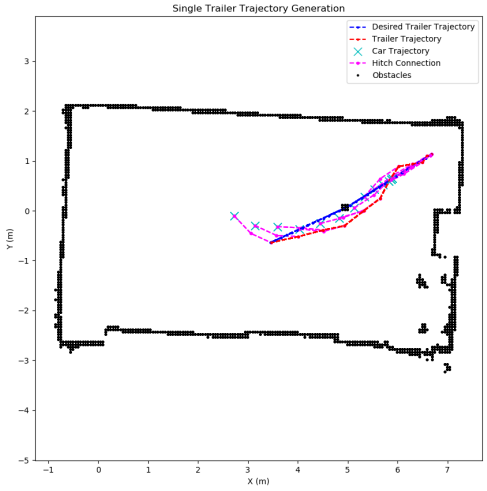

Fig. 5. Simulation trajectory results for 1 can

2) Experiment with 3 Cans: This experiment is similar to the previous one, but we added more cans to have a more complex trajectory. In this case, we have the car start at the top

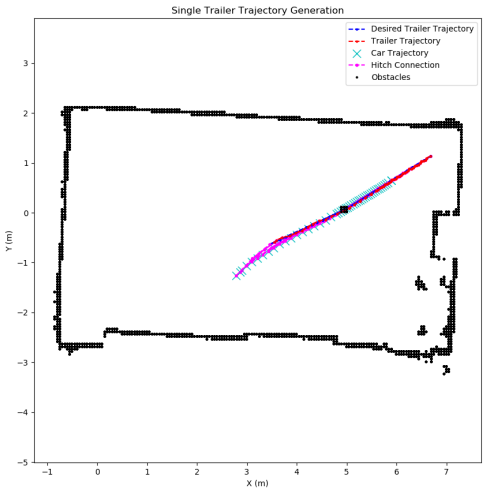

Fig. 6. Simulation trajectory results for 1 can without obstacle constraints

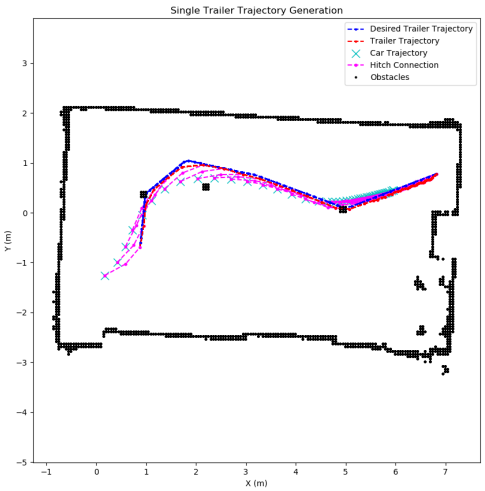

Fig. 8. Simulation trajectory results for 3 cans without obstacles constraints

right position and have a trajectory weave through three cans from right to left. We want to hit the first and third cans with the trailer and avoid the second can. The simulation output is shown in figure 7 and you might also notice that the trailer does not hit the third can, but we were satisfied with this result because we assumed the drift of the car in the real world would account this error. We also ran the same scenario without the obstacles constraint to make sure that the trajectory was plausible and cable of being followed; this is shown in figure 8.

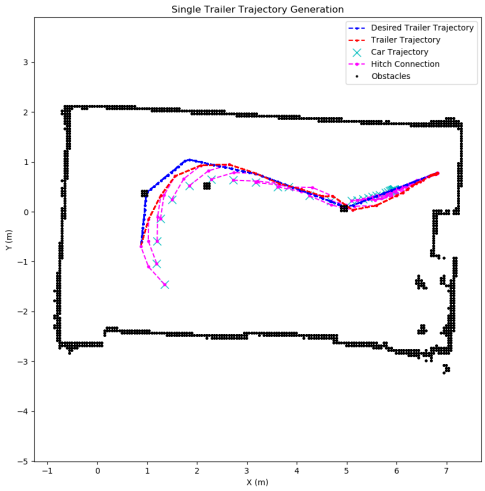

Fig. 7. Simulation trajectory results for 3 cans

C. Real World

1) Experiment 1 in the real world and tuning: We aimed to have the trailer pass through a single can without the car hitting it. This itself was a challenging task as with our trailer lengths and simple approximation of vehicle model would make it very tight. We tuned our optimization until we were satisfied with our results and got repeatable successes.

As expected the car did not follow the exact trajectory as in simulation and we had to tune a bit more. After a few trials we were able to get repeatable results and the video demonstration for the same can be found here. Figure (9) shows the trailer hitting the can.

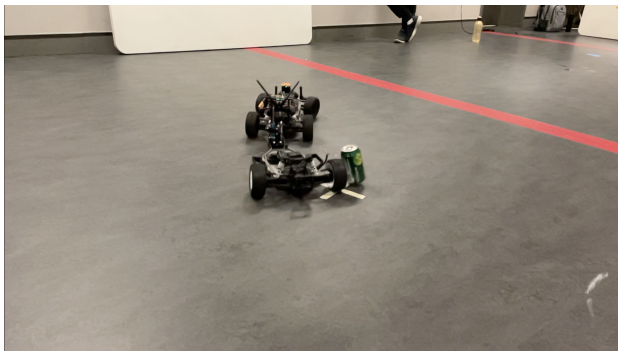

Fig. 9. Real World Experiment Setup for 1 can

2) Experiment 2 in the real world: We also performed the second experiment on the real car and tested the controller with multiple obstacles in the track and a complex path. This was a complex task that would test how well the trailer is controlled. We provided a trajectory that would pass through can 1 and 3 without passing through can 2. Evidently, this required more tuning, but we were also able to accomplish

this goal. The video demonstration for this experiment can be found here. Figure (10) shows the 3-trailer experiment setup.

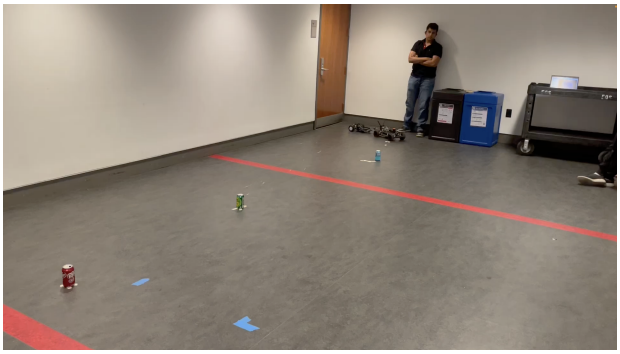

Fig. 10. Real World Experiment Setup for 3 can

3) Discussion: From the tests, we can see that the car's performance is very sensitive to initial conditions. This is reasonable as the offline planner accounts for the initial position and since the trajectory optimization **is not** done in real-time, any deviation from that will result in poor performance.

Note that to obtain the exact can position in the real world, we measured the position of the cans in RVIZ and then, as best we could, pinpointed those locations in the real world with a tape measure.

We also noticed that the car and trailer drift from the planned trajectory. Since this is not modeled in the dynamics, it has an effect on the real-world results. However, because of this, what seemed to be an unfulfilled result in simulation, resulted in a success.

VI. Challenges and Future Work

The project of designing and building an autonomous single-trailer car poses several challenges. One significant challenge we encountered was the gap between simulation and real-world scenarios. Although the system worked well in simulation, in real-world scenarios, the presence of drift due to the floor's friction, which was not modeled in our system dynamics, caused the system to miss or hit a can head-on instead of with the trailer. Additionally, the use of MPC for trajectory optimization offline, makes the system's performance very sensitive to its initial position. Hence, achieving repeatability during testing was difficult. Lastly, we found that MPC does not always solve and converge to a solution when more constraints are enforced, which limits the system's performance.

To improve the project's efficacy, future work can be done. One possible area of future work could be to further improve tuning and the objective function to hit the cans more precisely, ensuring that the car-trailer system can more accurately complete its task. Another potential area of future work could be to incorporate a reward function that provides a more robust and safer motion planning approach for the car when navigating around obstacles, improving the system's safety. Additionally, testing the system in more challenging can orientations and more constrained environments could help address some of the limitations of the current system. By addressing these challenges and implementing the suggested future work, we can make autonomous car-trailer systems more effective and efficient in real-world scenarios.

VII. Conclusion

In conclusion, we have demonstrated the feasibility and performance of an autonomous single-trailer car system that can safely and efficiently navigate through an environment with obstacles. By designing a control system using an MPC and Pure Pursuit, we were able to accurately track the desired car-trailer system trajectory. Through extensive testing and experimentation on different can positions, we have shown that the system can hit specific cans using the trailer while avoiding other cans, all while closely tracking the desired trailer trajectory. This shows the successful implementation of a control system for a single-trailer car system.

Overall, our project contributes to the development of the control of autonomous car-trailer systems and has the potential to benefit the logistics and transportation industry by improving efficiency, reducing costs, and increasing safety.

References

[1] Keymasi Khalaji, Ali, and S. Ali A. Moosavian. "Dynamic modeling and tracking control of a car with n trailers." Multibody System Dynamics 37.2 (2016): 211-225.

[2] Althoff, Matthias, Markus Koschi, and Stefanie Manzinger. "CommonRoad: Composable benchmarks for motion planning on roads." 2017 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2017.

[3] McDowell, Journey. "Comparison of modern controls and reinforcement learning for robust control of autonomously backing up tractor-trailers to loading docks." (2019).

[4] De Bernardis, Martino, et al. "On nonlinear model predictive direct yaw moment control for trailer sway mitigation." Vehicle System Dynamics (2022): 1-27.

[5] Khalil Alipour, et al. "Dynamics modeling and sliding mode control of tractor-trailer wheeled mobile robots subject to wheels slip". Mechanism and Machine Theory (2019): Vo. 138, pg. 16-37.

[6] Rouchon, Pierre, et al. "Flatness, motion planning and trailer systems." Proceedings of 32nd IEEE Conference on Decision and Control. IEEE, 1993.

[7] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas. Dynamic Programming and Optimal Control, volume 1. Athena Scientific Belmont, MA, 1995.

[8] E. F. Camacho and C. B. Alba. Model Predictive Control. Springer, 2013

[9] R. C. Coulter, "Implementation of the Pure Pursuit Path Tracking Algorithm," 1992, accessed: 017-04-2023. [Online]. Available: https://www.ri.cmu.edu/pub files/pub3/coulter r craig 1992 1/coulter r craig 1992 1.pdf