An Online Path Planning and Tracking System for Lane-keeping and Obstacle Avoidance

Xun Fu, Xi Lin, Bowen Mu, Hanyang Li
University of Michigan
Ann Arbor, USA

***Abstract*—Our project aims at designing an online path planning and tracking system for an autonomous vehicle to perform lane-keeping and obstacle avoidance. For path planning part, we implement two methods, artificial potential field and best first search, and compare their performance on some section of roads with different orientations and obstacle positions. The results show that the latter produces planned trajectories of higher quality, and its computation time satisfies online planning requirement, thus best first search is selected as path planning method for the system. For path tracking part, a multi-constraint model predictive control is constructed to calculate the steering angle to prevent the vehicle from colliding with obstacles in real-time. Real-time simulation results show that proposed path planning and tracking approach is effective. The codes of our project are posted on Github, the link is atta**

I. Introduction

Recently, a number of new technologies are empowering autonomous vehicles like never before, different safety systems have been developed and applied to improve driving safety, however, increasing density of transportation system makes the need of further reduction in traffic accidents indispensable. Also, changeable road conditions need path planning based on detected obstacles, and actuator control so that the vehicle can follow the path to avoid obstacles.

Based on best first search [4] and potential field algorithms [1], we develop two methods that can generate planned trajectories for vehicle control method to track, which satisfy both lane-keeping and obstacle avoidance requirements. We evaluate the quality of planned trajectories on sections road with different orientations and discover that best first search provide smoother trajectories and better lane-keeping performance than potential field, thus we select best first search as the path planning method. The method is fast to compute and can perform online planning.

Even though the collision-free trajectory can be generated, the kinematic model of the vehicle determines that there are physical constraints that can make the car deviate from the established trajectory. We must consider nonlinear characteristics of the vehicle. In this report, based on path tracking system proposed by Ji et al [2], we use multi-constrained model predictive control (MMPC) to systematically handle input constraints and admissible states.

The remainder of this report is organized as follows: Section II describes the path planning algorithm that generates a collision-free trajectory based on the artificial potential field and evaluation of its performance. Section III describes the path planning algorithm based on the best first search, and discusses its current problems and comparison with the algorithm of artificial potential field. Section IV presents the vehicle dynamic model used for path-tracking. Section V presents the detailed multi-constrained model predictive control (MMPC), including its design and analysis.

II. Path Planning with Artificial Potential Field

Before performing model predictive control, it's necessary to compute a planned trajectory for it to track. Our group tried two path planning methods, best first search and artificial potential field, which are introduced in this and the following section.

This section would talk about one of the way that using virtual potential field to implement the path planning algorithm for the collision avoidance system for the vehicle. This algorithm is implemented based on the 2D virtual world that provides the location of the obstacle and the information of the road. The idea of the algorithm is implementing a local path planning algorithm based on the virtual potential field. In the real world, the autonomous vehicles are always required to automatically navigate to the goal and avoid the obstacle in the environment. Artificial potential field (APF) provides a simple and elegant way for path planning.

A. Basic idea of potential field

The basic idea of the potential field is from the natural, such as a charged particle navigating a magnetic field. The basic idea is that suppose the goal is a point $g \in \mathcal{R}^2$, and the starting position is a point $r \in \mathcal{R}^2$. If there is no obstacle in the environment, the vehicle should automatically seeking the goal point. To achieve that, we would apply the suitable force that will drive the vehicle to the goal. We represent this force by applying an attractive potential field defined as U_{att}. The attractive potential field describes that, in each time step, we would calculate the potential field at current vehicle location and then calculate the induced force. The vehicle then could move according to this force.

In order to avoid the collision, for each of the obstacle, we could also define another force to push the vehicle away to avoid collision. We name this field repulsive potential field, U_{rep}. Then, the total potential field at time t could be represented by:

$$U_{total}(t) = U_{att}(t) + U_{rep}(t) \tag{1}$$

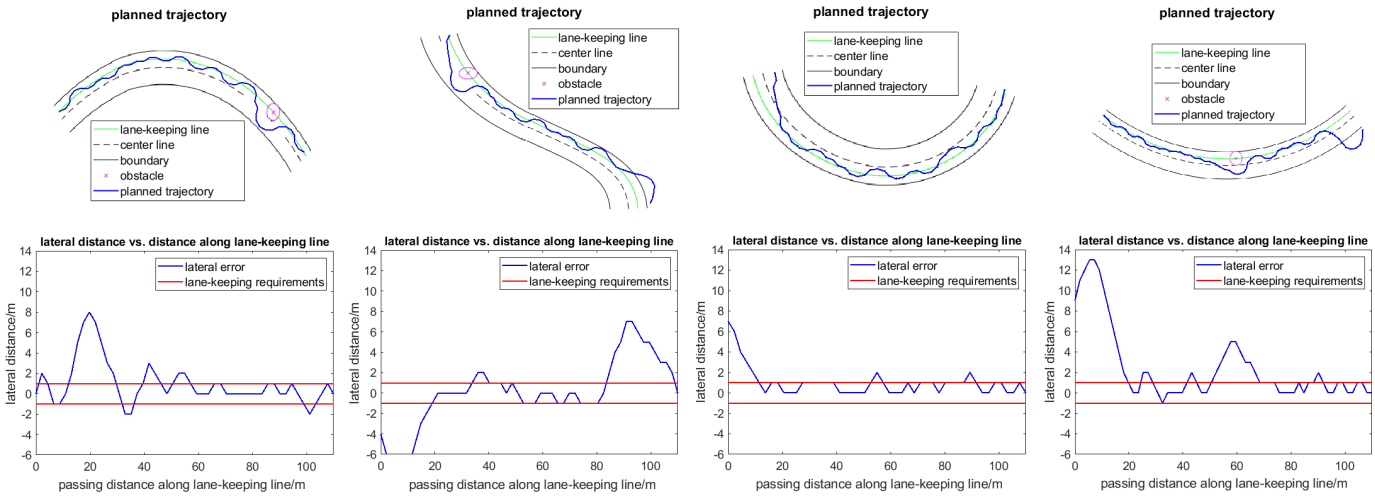

Fig. 1: Planned trajectories and lateral errors of potential field in different situations

B. implementation of algorithm

For the giving map and a fixed vehicle size, we would generate a matrix to represent the map. To simplify the derivation of the collision-free trajectory using potential field approach, we will assume following conditions:

- Only a single obstacle appears in front of the host vehicle in each step
- The obstacle would not move when vehicle moving
- There are boundaries exist on both side of the road

There are two coordinates of the map,(x,y). For each time step, the vehicle would calculate the Euclidean distance from the current position to the final destination. All obstacles repel the vehicle with a magnitude inversely proportional to the distance. The resultant potential, accounting for the attractive and repulsive components is measured and used to move the vehicle.

The distance of the obstacles at all angles from the vehicle is measured. In this algorithm, we use only 5 directions at specific angles are measured to compute the repulsive potential [3]. The angles are left side, right side, forward left diagonal and forward right diagonal. The MATLAB codes are posted on our github repository.

C. Evaluation of algorithm performance

We test the artificial potential field method on some arbitrary section of roads. Figure 1. shows the planned trajectories and their lateral distance from the lane-keeping line. The start positions of the vehicle vary within road width and the start orientation vary within $[-\frac{\pi}{3}, \frac{\pi}{3}]$ with respect to the road direction.

Artificial potential field motion planning algorithm provides a fast and elegant way for obstacle avoidance. However, there are a few problems by using this algorithm. Firstly, the vehicle would be limited inside of the boundaries of the road. In the project, in some situation, we may require our vehicle to move outside of the road to successfully avoid the collision. For this algorithm, we need to make some additional changes to make it happen, such as change the map of the road and add additional boundaries. This may make the system inefficiency and unstable because of changing the original data of the map. In additional, using this method, the vehicle can be easily fall in a local minimum. For example, when the vehicle went to an "U" shape corner, the vehicle may fail into the local minimum which inside the U shape, and even adding an addition force to get out of it, APF would still cost additional steps and time to pass around the local minimum.

III. Path Planning with Best First Search

A. Basic idea of best first search

Generally speaking, the best first search algorithm searches for the path by exploring neighbours of the point which has the lowest value according to heuristic function among all explored points at each step, and return the path when the goal is reached. In our work, the heuristic function is simply the distance from the query point to the goal as shown below.

$$h(p) = \sqrt{(p_x - goal_x)^2 + (p_y - goal_y)^2} \tag{2}$$

B. Incorporation of vehicle dynamic model

The best first search algorithm we use here is different from the standard version in terms of the definition of neighbour. While the standard one simply define neighbours of a point as points that surround it, it's not viable for vehicle path planning. The reachable area of a car at next instance not only depends on its current position, but also constrained by parameters such as vehicle velocity and heading angle. Hence, we utilize a simplified vehicle dynamic model shown below to take these factors into consideration.

$$\begin{cases} \dot{x} = u \cdot cos\theta \\ \dot{y} = u \cdot sin\theta \\ \dot{\theta} = \frac{u}{L} tan\phi \end{cases} \tag{3}$$

It should be mentioned that the model is only used to reduce computation complexity in path planning, a more realistic model is used in the tracking task. In the model,x,y and θ

reflect the position and heading angle of the car, u is the longitudinal velocity of the car, ϕ is the car's yaw angle, and L is the length of car. We assume that u is constant, then the only input to the model is yaw angle ϕ.

With the above model, we can compute the neighbours of a point that also consider the state of vehicle. Neighbours of a point are endpoints of motion primitives which are computed by giving a specific input to the model for some time. To search for path efficiently, we select an input set $\phi = \{-\frac{\pi}{10}, -\frac{\pi}{22}, 0, \frac{\pi}{10}, \frac{\pi}{10}\}$, and primitive time length to be 0.5 seconds. We discretize each primitive into 5 time steps, and compute the linear approximation of the segment using Euler integration and state variables from the last time step using the formula below.

$$\begin{cases} \dot{\theta} = \frac{u}{L} tan\phi \\ \theta_{k+1} = \theta_k + dt \cdot \dot{\theta} \\ x_{k+1} = x_k + dt \cdot u \cdot cos\theta_{k+1} \\ y_{k+1} = y_k + dt \cdot u \cdot sin\theta_{k+1} \end{cases} \tag{4}$$

C. Constrained search space design

The goal is to provide a planned trajectory that performs lane-keeping along the center line of the right lane as well as avoid obstacles, thus we design a constrained search space that is generated according to the actual vehicle state and obstacle position for the best first search algorithm to perform on. The search space is consist of three sections, namely entry, lane-keeping and obstacle avoidance section. Figure 2 shows what the sections look like and how the search algorithm works under different circumstances on a straight road, the initial position and orientation deviate slightly from the center line of right lane, appearing as a green line in the figure.

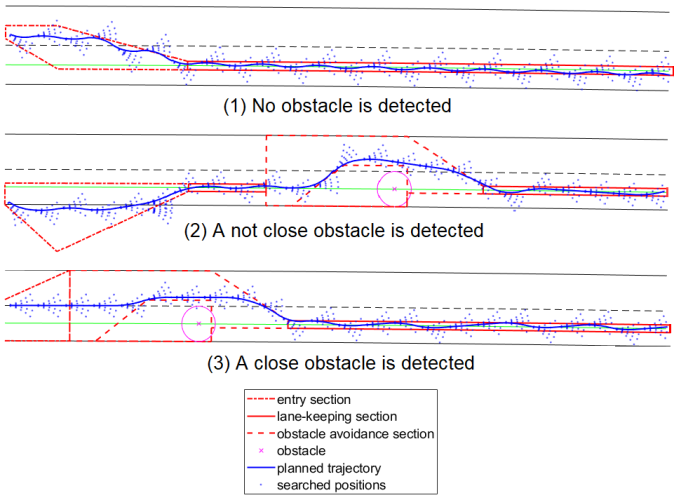

Fig. 2: Search space and planned trajectory in different situations

1) Entry section: The objective of this section is to move vehicle back to perform lane-keeping if the actual vehicle state deviates from the center line of right lane in the beginning. In the case when the initial position of vehicle is close to obstacle, this section will simply be the extension of obstacle avoidance section and don't lead the vehicle back to lane-keeping region. As shown in the graph above, the boundary of entry section starts near the initial position of vehicle with some margin, and its direction is designed to be proportional to the vehicle's initial orientation for a certain length, then converges towards center line of right lane. In this way, the search algorithm is offered some space to find a trajectory that is subjected to dynamic model and maneuvers smoothly towards lane-keeping region.

2) Lane-keeping section: This section's goal is the same as its name, and its boundary is also simple to understand, simply the line segments that are parallel to the center line of right lane with a certain lane-keeping tolerance clearance. It can be seen in the plots that this section usually follows the entry section, but can also be adjusted when an obstacle avoidance section presents.

3) Obstacle avoidance section: This section includes two part, one marks the region where obstacle presents and the trajectory searching is not allowed, appearing as a rectangular area with cone-shaped head, the other includes the whole width of road where our search algorithm can perform on, and its boundary converges to lane-keeping region just after the obstacle is passed.

D. Current limitation and problems

The method assumes that the vehicle can not make reverse movement, thus it would fail to plan the trajectory when the vehicle position can no longer move within search space by moving forward. Some currently found situations related to this issue are discussed here. First case is when the initial vehicle position is too close to an obstacle, but it should not be a problem if the planning frequency is not too low since an obstacle avoidance trajectory has already been planned before the vehicle is too close to an obstacle. Second case is related to the parameters specifying search space boundary. The values of these parameters in our codes actually reflects the number of road points, although most of the interval distances between points are approximately same, some sections of road have much denser road points, which can affect the shape and size of search space severely. Then, the planning may fail because of insufficient search space.

E. Evaluation of algorithm performance

We test the best first search algorithm on the same sections of road and vehicle start positions as artificial potential field, the results are shown in Figure 3. As shown in the trajectory plots, when the actual vehicle position deviate from the center line of right lane, the planned trajectory can steer it back to perform lane-keeping. When an obstacle is detected, the planned trajectory can steer the car to circumvent it and move back to the center line of right lane after that. It's also known that the lateral error of planned planned trajectory stay below the given lane-keeping requirements unless the vehicle deviates in the beginning or obstacle avoidance is necessary. It can be seen that even in cases where the vehicle's start position and orientation deviate seriously, this method can produce a satisfactory planned trajectory. Last but not least, our code

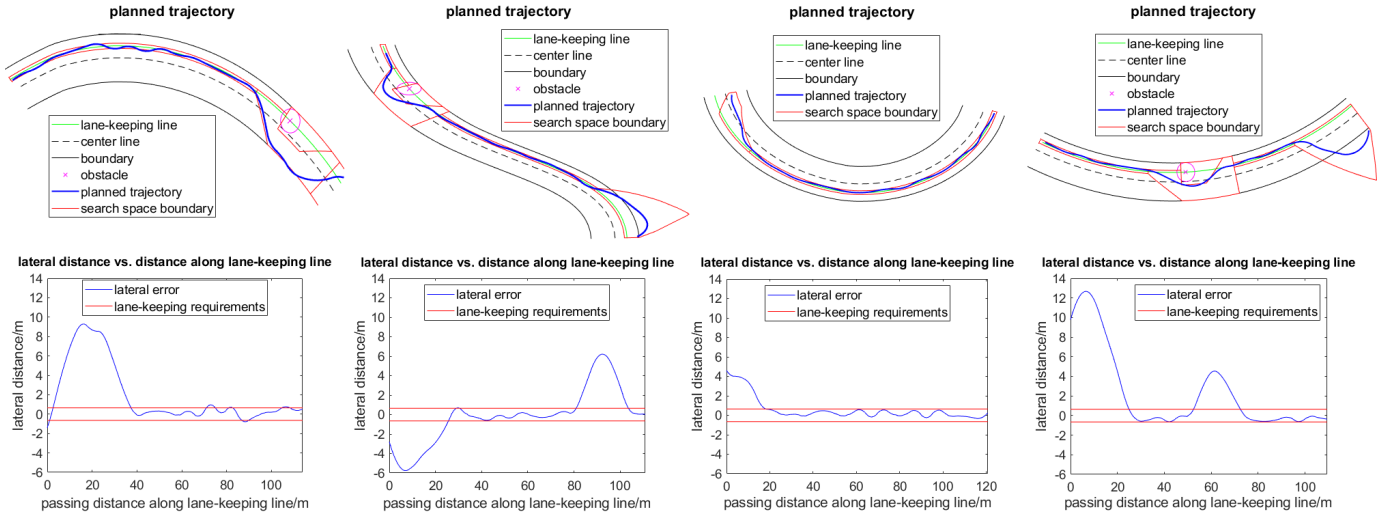

Fig. 3: Planned trajectories and lateral errors of best first search in different situations

F. Comparison with artificial potential field

Compared to artificial potential field, the path quality of best first search is obviously better. In terms of lane-keeping performance, The lateral error of trajectories obtained by the latter satisfies the requirements mostly, red lines in the error plots, while that of the former exceeds them more frequently. Besides, the latter produces much smoother trajectories than the former, which are more favorable for model predictive control method to track. Additionally, the latter is robust to different shape of road, and will not encounter problems like local minimum that causes difficulty to potential field. For these reasons, We select best first search as the path planning method in our work.

IV. Vehicle Mathematical Model for Path-Tracking Problem

For this project, we would use multi-constrained model predictive control(MMPC) for the path-tracking of the vehicle. The model we use for this project considers the aspect of kinematic and dynamic aspects of the vehicle. We will introduce this mathematical model of vehicle which used for collision avoidance. We will implement the MMPC on the discrete system and develops a vehicle dynamic model along with the lateral and yaw dynamics

A. Vehicle Dynamic Model for Path Tracking

For this project, we will assume a linearized vehicle model which the following assumption was made: 1) The longitudinal velocity of vehicle is constant 2) at the front and rear axles, the left, and right wheels are lumped in a single wheel and 3) suspension movements, slip phenomena, and aerodynamic influences are neglected. The model is shown in figure 4.

From this model, we could use the body side-slip angle β and the yaw rate of vehicle body $\dot{\psi}$ as state variables. Then the vehicle lateral dynamics can then be described as:

$$mV(\dot{\beta} + \dot{\psi}) = F_{xf} + F_{xr} \tag{5}$$

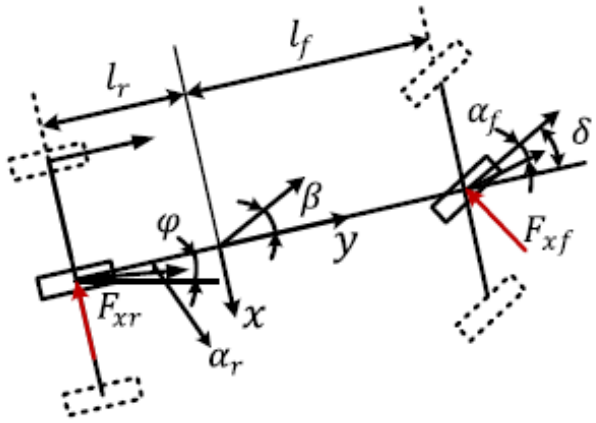

Fig. 4: vehicle dynamic model used for path tracking

$$I_z\ddot{\psi} = l_f F_{xf} - l_r F_{xr} \tag{6}$$

where I_s is the vehicle moment of inertia about the yaw axis and l_f and l_r are the longitudinal distance from the center of gravity(CG) of the vehicle to the front and rear wheels.

In addition, we would use some different models for the cornering tire force. In the project, we assume that for a small tire slip angle, the lateral tire forces are approximated as a linear function of tire slip angle. For the Front and rear tire forces F_{xf}, F_{xr} and the tire slip angle a_f, a_r are defined as :

$$F_{xf} = C_f \times a_f = C_f \times (\delta - \beta - \frac{l_f\dot{\psi}}{V}) \tag{7}$$

$$F_{xr} = C_r \times a_f = C_r \times (-\beta + \frac{l_r\dot{\psi}}{V}) \tag{8}$$

where δ is the front-wheel steering angle; C_f and C_r represent the linearized cornering stiffness of the front and rear

wheels. From previous equations, we could obtain the equation about the lateral and yaw dynamics in a vehicle model:

$$\dot{\beta} = \frac{-(C_r + C_f)}{mV}\beta + (\frac{(C_r l_r + C_f l_f)}{mV^2} - 1)\dot{\psi} + \frac{C_f}{mV}\delta \quad (9)$$

$$\ddot{\psi} = \frac{C_r l_f - C_f l_f}{I_z}\beta + \frac{(C_r l_r^2 + C_f l_f^2)}{I_z V}\dot{\psi} + \frac{C_f l_f}{I_z}\delta \quad (10)$$

B. Discrete Linear Vehicle Model for MPC

With the mathematical model we calculated, we could obtain the discrete state-space vehicle model. The vehicle state space model is formed by the lateral position of the vehicle CG X_v, and the vehicle side-slip angel β, the yaw angle ψ and the yaw rate $\dot{\psi}$. The input is given by the angle of front wheel δ. The state space vector is obtained as:

$$X_c = \begin{bmatrix} X_v & \beta & \psi & \dot{\psi} \end{bmatrix}^T \quad (11)$$

The state equation would be:

$$\dot{X}_c = A_c X_c + B_c \delta \quad (12)$$

$$Y_c = C_c X_c \quad (13)$$

where

$$A_c = \begin{bmatrix} 0 & V & V & 0 \\ 0 & -\frac{C_r + C_f}{mV} & 0 & \frac{C_r l_r + C_f l_f}{mV^2} - 1 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{C_r l_r + C_f l_f}{I_z} & 0 & -\frac{C_r l_r^2 + C_f l_f^2}{I_z V} \end{bmatrix}$$
$$B_c = \begin{bmatrix} 0 \\ \frac{C_f}{mV} \\ 0 \\ \frac{C_f l_f}{I_z} \end{bmatrix}, C_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (14)$$

Since we need to control the system in a discrete time system but the state-space model we just obtained is a continue time model. Therefore, we need to transfer it into discrete time state space vector as following:

$$X_d(k+1) = A_d X_d(k) + B_d u(k) \quad (15)$$

A_d and B_d are the state and control matrices for the discrete state-space equation which could be calculated by;

$$A_d = e^{A_c \Delta T} \quad (16)$$

$$B_d = \int_{k\Delta T}^{(k+1)\Delta T} e^{A_c[(k+1)\Delta T - \tau]} B_c d\tau \quad (17)$$

where ΔT is the sampling interval for the discrete state-space model.

The lateral displacement, side-slip angle, and yaw rate of vehicle are defined as output using:

$$Y_d(k) = C_d X_d(k) \quad (18)$$

where

$$Y_d(k) = \begin{bmatrix} X_v \\ \beta \\ \dot{\psi} \end{bmatrix}, C_d = C_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (19)$$

For path tracking by using MMPC, it is common to formulate the constrained control problem as a real-time optimization problem subject to hard constraints on plant variables and soft constraints on outputs. The discrete state-space model is augmented as an uniform model with integrator.

For discrete state variable, we know:

$$\Delta X_d(k+1) = X_d(k+1) - X_d(k) \quad (20)$$

$$\Delta u(k) = u(k) - u(k-1) \quad (21)$$

After substituting discrete state space model with upper equations, we could obtain the new model with increments of the variables $X_d(k)$ and $u(k)$ as

$$\Delta X_d(k+1) = A_d X_d(k) + B_d \Delta u(k) \quad (22)$$

$$Y_d(k+1) - Y_d(k) = C_d A_d \Delta X_d(k) + C_d B_d \Delta u(k) \quad (23)$$

Now, we could define a new state variable vector as

$$X_a(k) = \begin{bmatrix} \Delta X_d(k) \\ Y_d(k) \end{bmatrix} \quad (24)$$

Then we could obtain a new state space model with triplet (A_a, B_a, C_a) called augmented model, showed as following:

$$X_a(k+1) = A_a X_a(k) + B_a \Delta u(k) \quad (25)$$

$$Y_a(k) = C_a X_a(k) \quad (26)$$

where

$$A_a = \begin{bmatrix} A_d & O_a^T \\ C_d A_d & I_a \end{bmatrix}, B_a = \begin{bmatrix} B_d \\ C_d B_d \end{bmatrix}, C_a = \begin{bmatrix} O_a I_a \end{bmatrix},$$
$$O_a = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, I_a = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \quad (27)$$

V. Design Of Multiconstrained Model Predictive Control

In order to design an MPC for path tracking, we need to predict the future behavior of the vehicle at each time step. This future prediction determines the control inputs within a specified prediction horizon, and on the basis of these future states, a performance index is minimized to compute the optimal control inputs. In the project, we will assume the current time is k, $k > 0$; the prediction horizon is Np = 20 as the length of the optimization window and the control horizon

is Nc = 5. The state variable vector X_a(k) provides the current plant information, which is available through measurement. With given information X_a(k), the future state variables can be predicted for Np steps ahead, as follows:

$$X_a(k+1), ..., X_a(k+m), ..., X_a(k+N_p) \tag{28}$$

where $X_a(k+m)$ is the predicted state variable at k+m with given current plant information $X_a(k)$. By defining ΔU that represents the sequence of the future input increments computed at time k for the current observed states, that

$$\Delta U_m = [\Delta u(k), ..., \Delta u(k+m), ..., \Delta u(k+N_c-1)]^T \tag{29}$$

By using the ΔU_m and the state space (A_a, B_a, C_a) and the state variable are calculated by continuing iteration that:

$$\begin{aligned}
&X_a(k+1) = A_a X_a(k) + B_a \Delta u(k) \\
&X_a(k+2) = A_a^2 X_a(k) + A_a B_a \Delta u(k) + B_a \Delta u(k+1) \\
&... \\
&X_a(k+N_c) = A_a^{N_c} X_a(k) + A_a^{N_c-1} B_a \Delta u(k) + ... \\
&+ B_a \Delta u(k+N_c-1) \\
&... \\
&X_a(k+N_p) = A_a^{N_p} X_a(k) + A_a^{N_p-1} B_a \Delta u(k) + ... \\
&+ A_a^{N_p-B_c} B_a \Delta u(k+N_c-1)
\end{aligned} \tag{30}$$

Then we could define the state vector and predicted outputs for the predictive state-space model as

$$\begin{aligned}
&X_m(k) = X_a(k) \\
&= \begin{bmatrix} \Delta X_v(k) & \Delta\beta(k) & \Delta\psi(k) & \Delta\dot{\psi}(k) & X_v(k) & \beta(k) & \dot{\psi}(k) \end{bmatrix}^T
\end{aligned} \tag{31}$$

$$Y_m(k) = \begin{bmatrix} Y_a(k+1) & ... & Y_a(k+N_p) \end{bmatrix}^T \tag{32}$$

Then we could derive the prediction model of performance outputs over the prediction horizon N_p in a compact matrix form as

$$Y_m(k) = F_m X_m(k) + G_m \Delta U_m \tag{33}$$

where

$$\begin{aligned}
F_m &= \begin{bmatrix} C_a A_a & C_a A_a^2 & ... & C_a A_a^{N_c} & ... & C_a A_a^{N_p} \end{bmatrix}^T_{3N_p \times 7} \\
G_m &= \begin{bmatrix} C_a B_a & 0 & ... & 0 \\ C_a A_a B_a & C_a B_a & ... & 0 \\ \vdots & & & \\ C_a A_a^{N_c-1} B_a & C_a^{N_c-2} B_a & ... & C_a B_a \\ \vdots & & & \\ C_a A_a^{N_p-1} B_a & C_a^{N_p-2} B_a & ... & C_a A_a^{N_p-N_c} B_a \end{bmatrix}_{3N_p \times N_c}
\end{aligned} \tag{34}$$

A. Development of Cost Function With Vehicle Dynamics

Based on the planned trajectory $P_r(k)$, sideslip angle $\beta_r(k)$ and yaw rate $\dot{\psi}_r(k)$ at time k which are chosen as the given set-point information for MMPC,We define the cost function J_E that reflects the control objective as:

$$\begin{aligned}
J_E = &[P_r(k) - P_v(k)]^T [P_r(k) - P_v(k)] + [\beta_r(k) - \beta_v(k)]^T \\
&\times [\beta_r(k) - \beta_v(k)] + [\dot{\psi}_r(k) - \dot{\psi}_v(k)]^T [\dot{\psi}_r(k) - \dot{\psi}_v(k)] \\
&+ \Delta U_m^T \bar{R} \Delta U_m
\end{aligned} \tag{35}$$

where, as in MMPC notation, P_v(k), β_v(k), and $\dot{\psi}_v$(k) are the predicted sequence of lateral position, sideslip angle, and yaw rate of vehicle in the fixed earth coordinate system.

Consider the autonomous vehicle traveling with constant longitudinal velocity V on the planned trajectory, and assumme that the curvature of the planned trajectory is C_T at instant time k. The reference sideslip angle of the vehicle is automatically determined by

$$\beta_{curv} = C_T (L_r - \frac{L_f m V^2}{2 C_r L}) \tag{36}$$

$$C_T = \frac{\mid \frac{d^2 X_t}{dY_T^2} \mid}{[1 + (\frac{dX_t}{dY_T})^2]^{\frac{3}{2}}} \tag{37}$$

where L = $L_f + L_r$ is used to denote the wheelbase of the vehicle; X_T , Y_T are the coordinates of the planned trajectory in the fixed earth coordinate system.

We need to limit β_{curv} in the interval $[-\beta_{max}, \beta_{max}]$ where

$$\beta_{max} = \begin{cases} 2\dfrac{k_1 - k_2}{V_r^3} - 3\dfrac{k_1 - k_2}{V_r^2} V^2 + k_1, & if V < V_r \\ k2, & if V \geq V_r \end{cases} \tag{38}$$

and V_r is the characteristic speed. In the project, we will choose k1 and k2 as $\pi/18$ and $\pi/60$

Then we define the reference signal of side-slip angle as

$$\beta_r(k) = \begin{cases} \beta_{max}, & \beta_r(k) > \beta_{max} \\ \beta_{curv}, & \beta_r(k) \in [-\beta_{max}, \beta_{max}] \\ -\beta_{max}, & \beta_r(k) < -\beta_{max} \end{cases} \tag{39}$$

Define the yaw rate $\dot{\psi}_{curv}$ the desired orientation of autonomous vehicle as

$$\dot{\psi}_{curv} \approx V \cdot C_T \cos \beta_{curv}. \tag{40}$$

For the lateral stability of the vehicle, a constraint of the maximum yaw rate $\dot{\psi}_{max}$ is defined, using the friction coefficient μ, gravity g, and the longitudinal velocity V , as

$$\dot{\psi}_{max} = \frac{\mu g}{V} \tag{41}$$

then we define the reference signal of yaw rate as

$$\dot{\psi}_r(k) = \begin{cases} \dot{\psi}_{max}, & \dot{\psi}_r(k) > \dot{\psi}_{max} \\ \dot{\psi}_{curv}, & \dot{\psi}_r(k) \in [-\dot{\psi}_{max}, \dot{\psi}_{max}] \\ -\dot{\psi}_{max}, & \dot{\psi}_r(k) < -\dot{\psi}_{max} \end{cases} \tag{42}$$

B. Constraint Analysis for MPC

For the MPC path tracking, we have three major types of constraints. The first two deal with constraints imposed on the control variables, and the third deal with the output constraint.

According to the kinematics and dynamics of vehicle model, the constraints, which are imposed on steering angle and stability, for the path-tracking problem are specified as

$$-C_1\Delta\delta^{max} \leq \Delta U_m \leq C_1\Delta\delta^{max} \tag{43}$$

$$-C_1\Delta\delta^{max} \leq C_1u(k-1) + C_2\Delta U_m \leq C_1\Delta\delta^{max} \tag{44}$$

$$C_3\begin{bmatrix} X^{min} \\ -\beta^{max} \\ -\dot{\psi}^{max} \end{bmatrix} \leq F_mX_m(k) + G_m\Delta U_m \leq C_3\begin{bmatrix} X^{max} \\ \beta^{max} \\ \dot{\psi}^{max} \end{bmatrix} \tag{45}$$

where δ^{max} and $\Delta\delta^{max}$ are the constraints of inputs, and X_{max} and X_{min} are the coordinate values of left and right boundaries of road in X-direction, Thus

$$\begin{aligned} C_1 &= \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix}_{N_c\times 1}, C_2 = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \\ 1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & \dots & 1 & 1 \end{bmatrix}_{N_c\times N_c} \\ C_3^T &= \begin{bmatrix} 1 & 0 & 0 & \dots & 1 & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 1 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 & 1 \end{bmatrix}_{3N_F\times 3} \end{aligned} \tag{46}$$

subject to the inequality constraints

$$\begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix}\Delta U_m \leq \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix} \tag{47}$$

The data matrices are given by:

$$\begin{aligned} M_1 &= \begin{bmatrix} -C_2 \\ C_2 \end{bmatrix}, M_2 = \begin{bmatrix} -I \\ I \end{bmatrix} M_3 = \begin{bmatrix} -G_m \\ G_m \end{bmatrix} \\ N_1 &= \begin{bmatrix} C_1\delta^{max} + C_1u(k-1) \\ C_1\delta^{max} - C_1u(k-1) \end{bmatrix} N_2 = \Delta\delta^{max}\begin{bmatrix} C_1 \\ C_1 \end{bmatrix} \\ N_3 &= \begin{bmatrix} -C_3\begin{bmatrix} X^{min} \\ -\beta^{max} \\ -\dot{\psi}^{max} \end{bmatrix} + F_mX_m(k) \\ C_3\begin{bmatrix} X^{max} \\ \beta^{max} \\ \dot{\psi}^{max} \end{bmatrix} - F_mX_m(k) \end{bmatrix}. \end{aligned} \tag{48}$$

C. Numerous Solution Using Hildreth's Quadratic Programming Procedure

The next step is to calculate the future steering inputs that optimize the path-tracking response of the vehicle. For this project, we suppose the objective function J_E and the constraints, as follows:

$$J_E = \frac{1}{2}\Delta U_m^T E_m\Delta U_m + \Delta U_m^T F_m \tag{49}$$

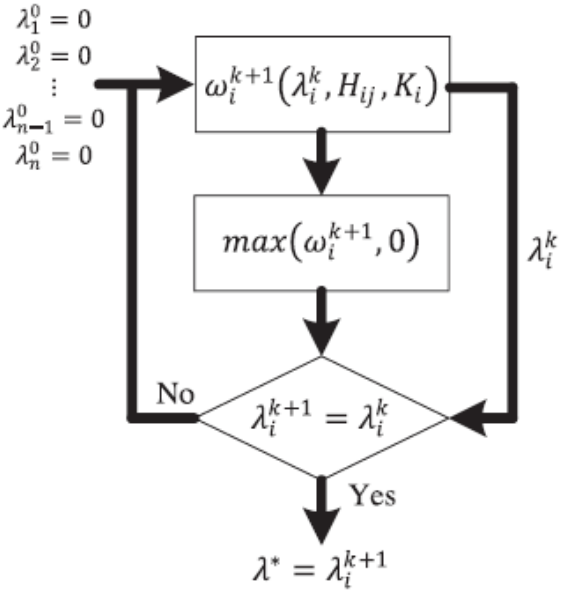

Fig. 5: Hildreth's quadratic programming for MMPC

$$M_m\Delta U_m \leq N_m \tag{50}$$

where $M_m = [M_1M_2M_3]^T$, $N_m = [N_1N_2N_3]^T$, and E_m, F_m are compatible matrices and vectors in the quadratic programming problem, as follows:

$$E_m = 2(G_m^TG_m + \overline{R}) \tag{51}$$

$$F_m = -2G_m^T[R_r - F_mX_m(k)]. \tag{52}$$

To minimize the objective function subject to inequality constraints, we consider the Lagrange expression

$$J_L = \frac{1}{2}\Delta U_m^T E_m\Delta U_m + \Delta U_m^T F_m + \lambda^T(M_m\Delta U_m - N_m). \tag{53}$$

The dual problem to the original primal problem could be derived as

$$\max_{\lambda\geq 0}\min_{\Delta U_m}(J_L). \tag{54}$$

From the first derivative of the cost function J_L, the minimization over ΔU_m is unconstrained and is attained by:

$$\Delta U_m = -E_m^{-1}(F_m + M_m^T\lambda). \tag{55}$$

After substituting the previous two equations, the problem becomes a quadratic programming problem with λ as the decision variable.The dual problem is written as :

$$\min_{\lambda\leq 0}(\frac{1}{2}\lambda^TH_m\lambda + \lambda^TK_m\lambda + \frac{1}{2}\lambda^TE_m^{-1}\lambda) \tag{56}$$

where matrices H_m and K_m are given by

$$H_m = M_mE_m^{-1}M_m^T \tag{57}$$

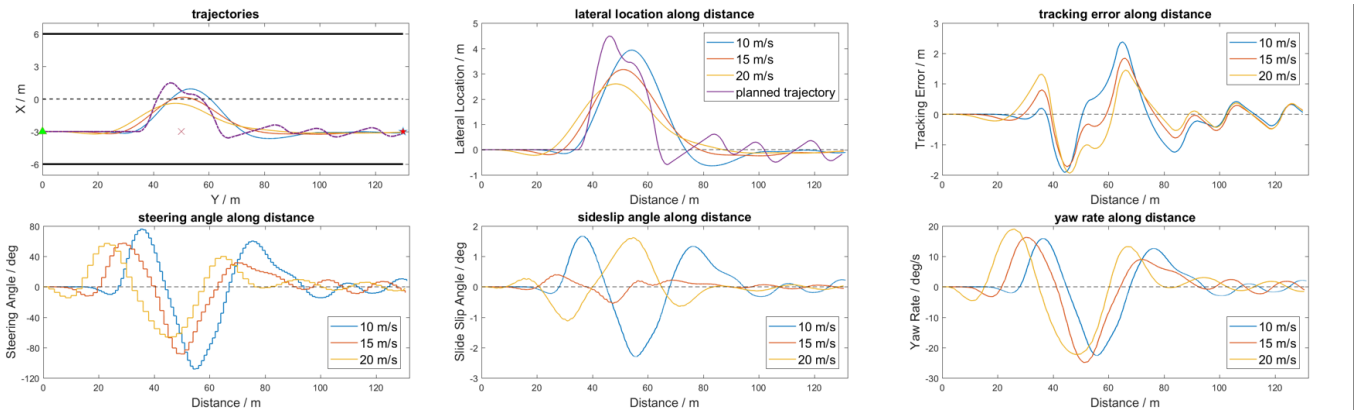

Fig. 6: Planned trajectories and lateral errors of best first search in different situations

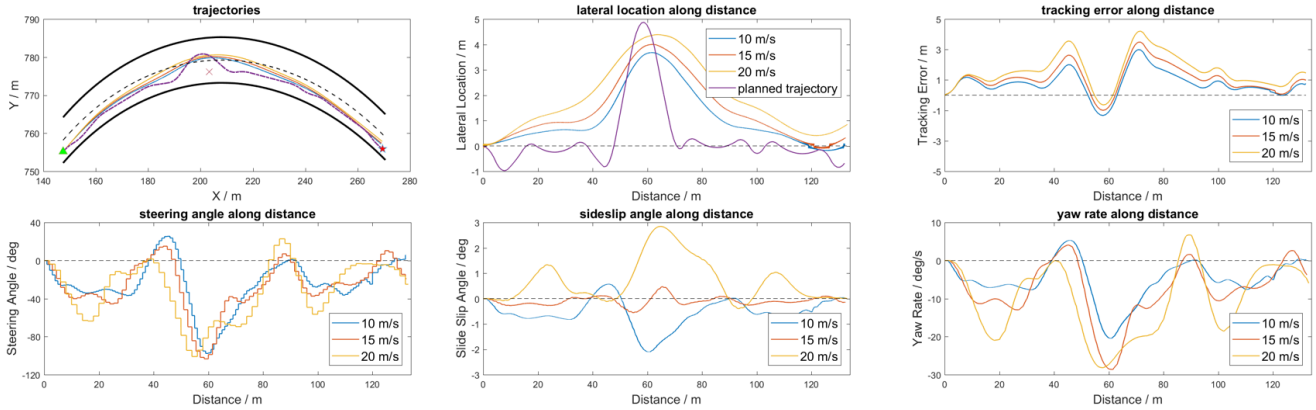

Fig. 7: Planned trajectories and lateral errors of best first search in different situations

$$K_m = N_m + M_m E_m^{-1} F_m. \tag{58}$$

This Hildreth's quadratic programming procedure was adopted for solving the optimal Lagrange multipliers that minimize the dual objective function. We will adjust a single component λ_i in vector λ^* to minimize the objective function. We will replace the decision λ in the ΔU_m equation that:

$$\Delta U_m = -E_m^{-1}(F_m + M_m^T \lambda^*). \tag{59}$$

VI. SIMULATIONS OF PATH TRACKING USING CARSIM AND SIMULINK

To verify the correctness and investigate the performance of the proposed work, simulations with MPC have been conducted using vehicle simulation software, i.e., CarSim, and MATLAB/Simulink.

The simulations represent an obstacle avoidance and lane keeping maneuver, in which the vehicle is following the planned trajectory with a constant speed. The control input is the steering angle, and the goal is to avoid the obstacle and follow the trajectory as close as possible with smooth enough steering angle input.

Section VI-A describes details of the simulation scenarios and configurations, and Section VI-B presents the simulation results and analysis.

A. Scenarios Description and Simulation Configurations

We are going to apply proposed work to two different scenarios. In the first scenario, suppose that vehicle is moving on a straight road of two lanes with constant velocity (10m/s, 15m/s,20m/s), in which there exist an obstacle located on the centerline of right lane. The vehicle is supposed to track the trajectory that is generated by using the method proposed in Section III to avoid the obstacle. In the second scenario, vehicle is moving on a low curvature curve road.

The sampling time in the simulations of both path planning and path tracking are 0.1 s. Prediction horizon and control horizon of MPC are 20 and 5 respectively. Simulations are conducted in real time.

B. Simulation Results

Symbol	Description	Value [units]
m	Total vehicle mass	1531 [kg]
I_z	Yaw moment of inertia	2347 [kg·m^2]
V	Velocity of vehicle	10,15,20 [m/s]
l_f	C.g. distance to front wheels	1.10 [m]
l_r	C.g. distance to rear wheels	1.68 [m]
C_f	Front wheel cornering stiffness	1200 [N/deg]
C_r	Rear wheel cornering stiffness	940 [N/deg]

TABLE I: Vehicle Model Parameters

TABLE I [2] defines the vehicle model parameters.

1) Scenario 1: Figs.6 show the performance of MPC-based path tracking controller applied to the vehicle that is moving on a straight road. Fig.6 shows three vehicle trajectories with different three constant velocities and the corresponding planned trajectory. As can be seen, vehicle does try to track to the path to avoid the obstacle and back to the centerline of the right lane after passing the obstacle. Fig.6 shows the tracking errors between vehicle actual trajectory and planned trajectory, the largest tracking error is about 2m. In Figs.6 and 6, we observe that vehicle avoids the obstacle with relatively low level of yaw rate and sideslip angle. The steering angle, yaw rate and sideslip angle are smooth due to the existence of state constraints and consideration of steering angle input in cost function.

2) Scenario 2: Figs.7 show the performance of MPC-based path tracking controller applied to the vehicle that is moving on a low curvature road. As can be seen, the proposed controller can be applied to the scenario in which the curvature of road is relative smooth and low. The goal of obstacle avoidance and lane keeping are also realized in this scenario. We observe that the proposed simulation results' tracking error can be decreased by tuning parameters of MPC. The proposed simulation results are generated under the condition in which the weighting matrix associated with the steering input in cost function is set to be approximately equal to the weighting matrix associated with tracking error. That means the solution of QP problem is doing some trade offs between less tracking errors and smoother, lower level steering angle input.

VII. Conclusion

In this report, we have presented a framework for path planning and path tracking for a collision avoidance system of autonomous vehicles. A real-time collision-free trajectory was generated for path tracking. For path-tracking, a real-time multi-constraints model predictive control method was proposed to control the vehicle. The state constraints on lateral position, yaw rate, and side-slip angle and the input constraint on the steering wheel angle were proposed to stabilize the vehicle at high speeds. It is solved with Hildreth's quadratic programming procedure, and the constraints were incorporated in an augmented vehicle model. Our code link is https://github.com/bexilin/ME_561_final_project.

References

[1] Ding Fu guang; Jiao Peng; Bian Xin-qian; Wang Hong-jian. Auv local path planning based on virtual potential field, mechatronics and automation. volume Vol.4, pages 1711–1716, 2005.

[2] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang. Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. *IEEE Transactions on Vehicular Technology*, 66(2):952–964, Feb 2017.

[3] R. Kala. Robot path planning using artificial potential fields. 2014.

[4] J. Petereit, T. Emter, C. W. Frey, T. Kopfstedt, and A. Beutel. Application of hybrid a* to an autonomous mobile robot for path planning in unstructured outdoor environments. In *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6, May 2012.