Autonomous Vehicle Parking in Dynamic Environments: An Integrated System with Prediction and Motion Planning

Leu, Jessica; Wang, Yebin; Tomizuka, Masayoshi; Di Cairano, Stefano

TR2022-056  June 03, 2022

Abstract

This paper presents an integrated motion planning system for autonomous vehicle (AV) parking in the presence of other moving vehicles. The proposed system includes 1) a hybrid environment predictor that predicts the motions of the surrounding vehicles and 2) a strategic motion planner that reacts to the predictions. The hybrid environment predictor performs short-term predictions via an extended Kalman filter and an adaptive observer. It also combines short-term predictions with a driver behavior cost-map to make long-term predictions. The strategic motion planner comprises 1) a model predictive control-based safety controller for trajectory tracking; 2) a search-based retreating planner for finding an evasion path in an emergency; 3) an optimization-based repairing planner for planning a new path when the original path is invalidated. Simulation validation demonstrates the effectiveness of the proposed method in terms of initial planning, motion prediction, safe tracking, retreating in an emergency, and trajectory repairing.

IEEE International Conference on Robotics and Automation (ICRA) 2022
Autonomous Vehicle Parking in Dynamic Environments: An Integrated System with Prediction and Motion Planning

Jessica Leu, Yebin Wang, Masayoshi Tomizuka, and Stefano Di Cairano

Abstract—This paper presents an integrated motion planning system for autonomous vehicle (AV) parking in the presence of other moving vehicles. The proposed system includes 1) a hybrid environment predictor that predicts the motions of the surrounding vehicles and 2) a strategic motion planner that reacts to the predictions. The hybrid environment predictor performs short-term predictions via an extended Kalman filter and an adaptive observer. It also combines short-term predictions with a driver behavior cost-map to make long-term predictions. The strategic motion planner comprises 1) a model predictive control-based safety controller for trajectory tracking; 2) a search-based retreat-planning planner for finding an evasion path in an emergency; 3) an optimization-based repairing planner for planning a new path when the original path is invalidated. Simulation validation demonstrates the effectiveness of the proposed method in terms of initial planning, motion prediction, safe tracking, retreating in an emergency, and trajectory repairing.

I. INTRODUCTION

Fully autonomous parking [1], [2] remains challenging especially in a dynamic environment with multiple independent agents, not only because it involves motion planning in a tight space, but also because autonomous vehicles (AVs) should intelligently react to the surrounding vehicles, i.e., obstacle vehicles (OVs). In contrast to driving on roads or highways, vehicle motions in parking areas do not have a clear set of rules to follow and largely depend on the driver’s intention or even skill level. These make the environment prediction in parking challenging, hence an autonomous parking system that integrates both prediction and planning is possibly necessary.

Motion prediction is crucial because it determines the safety constraints of the planning modules and thus the feasibility and smoothness of the motion plan [3]. Particularly, an accurate short-term motion prediction enables the AV to plan and react safely to the OVs; whereas long-term plan/mode prediction allows the AV to plan more efficiently and smoothly. This work proposes a model-based hybrid predictor to perform both short-term motion and long-term mode predictions by observing the poses of OVs. A major challenge in short-term prediction is to estimate the OV’s steering angle. We use extended Kalman filter (EKF) to reconstruct OV’s velocity, and then, resort to an adaptive observer for the steering estimation. Despite of the difficulty in predicting exact long-term motions, we observe that a driver generally follows some routes as a result of driving conventions (e.g., cars should stay on their left hand side in Japan). Also, vehicles’ motion throughout parking/leaving can be captured by several “modes” (e.g., maneuvering into/out of tight-spaces and cruising on aisles). Based on these two priors, we use a cost-map [4], [5] to capture these routes, combine the short-term predictions to determine OVs’ modes, and make long-term predictions.

Motion planning for AVs is another challenge in parking scenarios. General motion planning algorithms [6]–[11] are not directly suitable for parking in the presence of OVs which requires rapid replanning for complicated driving maneuvers. On the other hand, motion planners specialized in autonomous parking either fail to integrate short-horizon planning with long-horizon planning or cannot incorporate online path repairing upon new obstacles in dynamic environments [2], [12]–[16]. A scenario-aware planner that implements multiple strategies can be effective in terms of computation time, leading to a high replanning rate, and possibly safety guarantees. In this work, we first generate a long-term motion reference with Bi-Directional A-Search Guided Tree (BIAGT) [17]. Then, a strategic motion planner, based on the results of the hybrid environment predictor, implements three strategies: 1) model predictive control (MPC)-based safety controller [18] for trajectory tracking if the reference remains valid regarding the environment, 2) search-based retreat-planning that quickly finds an evasion path in an emergency, and 3) optimization-based repairing planning when the reference is invalidated.

This work presents an integrated system for autonomous vehicle parking in dynamic environments. Main contributions are threefold as follows:

- A model-based hybrid environment predictor predicts short-term motions and long-term modes.
- A strategic motion planner is presented to efficiently plan under different situations.
- Simulation is performed to show the effectiveness of the proposed system.

II. RELATED WORKS

A. Predictor

Research in vehicle motion prediction has attracted a lot of interests, and results in numerous contributions, e.g., short-term motion prediction methods [19]–[21] and long-term
plans/modes prediction methods [20]–[26]. It is arguably true that most research in this area is related to road driving. Interested readers are referred to an extensive survey in [27]. In contrast, vehicle motion prediction in parking is less explored. In [28], an interacting multiple model (IMM) filter is used to predict short-term trajectories in parking. Focusing on long-term prediction, [29] first trains a trajectory cluster classifier, and then acquires the mean-value trajectory of the classified cluster. In [24], the classified driver’s intent and the vehicle’s pose history are used to generate short-term motion predictions with a Long Short-Term Memory network. Purely data driven methods are not desirable for two reasons: 1) the lack of guarantees; 2) their performance depends largely on the training data set, and they may present a larger prediction error if the data set is chosen poorly. Prediction network over fitting may also be a concern. To the best of our knowledge, there are not extensive studies on a predictor fusing both short-term and long-term predictions for parking.

B. Planning

Prevailing motion planning approaches fall into three categories: search-based [30]–[32], sampling-based [33]–[35] and optimization-based [36]–[39]. Sampling-based planners could raise concerns in risk-sensitive tasks due to their non-deterministic nature, while optimization-based planners are only locally optimal and often need to work with global planners [39]–[43]. Various search-based motion planners are widely adopted by autonomous vehicles for their computation efficiency with well-chosen motion primitives and heuristics [6], [44]–[50].

If the parking environment changes, the initial long-term trajectory may need to be repaired. Real-time trajectory repairing methods include online heuristic update [51]–[53], pruning and reconnecting sampling-based search structures [54]–[57], and spline-based kinodynamic search [58], [59]. The heuristic update method is not directly applicable to our tree-based search structure in BIAGT and pruning is less efficient for parking scenarios. Spline-based kinodynamic search is relatively efficient but the original trajectory is not utilized. On the other hand, we observe that alternative feasible solutions in the repairing scenarios are often in the same homotopy class as the original trajectory. Therefore, optimization-based methods [37], [58], [60] become suitable candidates for repairing an existing path.

C. Parking system

As for system-level strategies for AV parking, [61] and [62] present autonomous parking systems to park in static environments. In [28], IMM is used for prediction with a sampling-based method for planning. The method in [22] first predicts the strategy of OVs, and then selects the navigation strategy of the ego AV. However, the AV only traverses the roads of the parking lot but do not perform parking maneuvers. Instead, our work aims at the more complete approach of an integrated parking system that makes both short-term and long-term predictions of the environment and utilize them in the strategic planning for parking.

Fig. 1: The simplified bicycle vehicle model. $L$ is the distance between the axis of the rear wheels and the axis of the front wheels.

Fig. 2: The integrated prediction and planning system.

III. METHODS

A. Problem Statement

Consider the planning problem with vehicle dynamics:

$$\dot{X} = f(X) + g(X, u),$$

where $X = [x, y, \theta]^T$ denotes the 2D coordinates and the vehicle heading, and $u = [\delta, v]^T$ is the control input that includes longitudinal velocity and steering angle. A collision-free configuration space $C_{free} \subset \mathbb{R}^n$ is the set of configurations at which the vehicle has no intersection with the obstacles. The motion planning problem considered in this paper is defined as follows:

Problem 3.1: Given an initial configuration $X_0 \in C_{free}$, a goal configuration $X_f \in C_{free}$, and system (1), find a feasible trajectory $P_t$ which

(I) starts at $X_0$ and ends at $X_f$, while satisfying (1); and (II) lies in the collision-free configuration space $C_{free}$.

We use the bicycle model, illustrated by Fig. 1, to represent the vehicle motion. The discrete-time model is obtained through Euler discretization as follows:

$$\begin{bmatrix}
    x_{k+1} \\
    y_{k+1} \\
    \theta_{k+1}
\end{bmatrix} =
\begin{bmatrix}
    x_k \\
    y_k \\
    \theta_k
\end{bmatrix} +
\begin{bmatrix}
    v_k T_s \cos(\theta_k) \\
    v_k T_s \sin(\theta_k) \\
    v_k T_s \tan(\delta_k) / L
\end{bmatrix},$$

where $T_s$ is the sampling time.

B. Proposed Architecture

Fig. 2 shows the architecture of the proposed system. The two main components are the hybrid environment predictor (Section III-C) and the strategic motion planner (Section III-D). During run-time, the central control first processes the
The hybrid environment predictor (summarized in Alg. 1) contains three main parts: motion estimation, motion prediction, and mode estimation of OVs.

1) Cascaded motion estimation: Motion estimation has been studied in, e.g. [24], [28], where the state $X$ is reconstructed from the measurement of $(x, y)$ based on the unicycle model. Such a treatment is not sufficient for vehicle parking, where frequent changes of moving direction and steering actions are involved. To accurately predict the short-term motion of the OV, it is advantageous to reconstruct the control input $u$. One can either pose it as an unknown input estimation problem [63] or augment the OV’s system state with the control input and solve a state estimation problem. Assume that the OV motion evolves according to the model (2). We obtain the augmented model of the OV by assuming that control input $(\delta, v)$ are piecewise constant, and estimate the augment state $[x, y, \theta, \delta, v]^T$.

Given the nonlinear augmented model, it is natural to apply well-established nonlinear state estimators such as EKF or particle filter for state estimation. We observe that it is not straightforward to tune EKF for accurate estimation of the steering angle. This is partially attributed to the term $v \tan(\delta)$ which involves the multiplication of unmeasured states. Meanwhile, the heavy computation presents a hurdle for the adoption of particle filter.

A cascaded motion estimator (Fig. 3) is proposed to estimate the OV motion. The velocity estimation resorts to EKF and is based on the following discrete-time model:

$$
\begin{bmatrix}
    x_{k+1} \\
    y_{k+1} \\
    \theta_{k+1} \\
    v_{k+1} \\
    \omega_{k+1}
\end{bmatrix} =
\begin{bmatrix}
    x_k + u_k T_s \cos(\theta_k) \\
    y_k + u_k T_s \sin(\theta_k) \\
    \theta_k + T_s \omega_k \\
    v_k \\
    \omega_k
\end{bmatrix} + q_k,
$$

where $\omega$ is the yaw rate, $q$ is the disturbance, $z_k = [x_k, y_k, \theta_k]^T$ is the measurement, $X_{cc} = \begin{bmatrix} x, y, \theta, \delta, v \end{bmatrix}^T$ is the augment state, $X_{EKF} = EKF$ and is based on the following discrete-time model:

$$
X_{EKF} = \begin{bmatrix} x, y, \theta, \delta, \omega \end{bmatrix}^T
$$

$$
X_{cc} = \begin{bmatrix} x, y, \theta, \delta, v \end{bmatrix}^T
$$

2) Short-term motion prediction: For the sake of computation efficiency, we assume the short-term motion of an OV is fully captured by the mean value of the state $X_{cc}$, and its covariance. For the mean value, we propagate the estimated states $X_{cc}$ forward and obtain a short-term prediction $X_{H,k} = [X_{1,k}^T, \ldots, X_{H,k}^T]^T$ for the future $H$ steps of the time horizon. Similarly, forward propagation is carried out to obtain the covariance matrices $P_{H,k} = \{P_{m_{H,k}}, \ldots, P_{m_{H,k+H}}\}$ according to EKF’s forward prediction formula. These information will facilitate long-term prediction and be used to determine the safety margin for each future time step.

3) Long-term mode prediction: OV’s long-term motion is dependent on the history of its state, the dynamic model, and its relative movement against the environment, where the first two factors are captured to some extent by the short-term motion prediction. In order to exploit the relative movement against the environment, [4] and [5] introduced a cost map to capture an OV’s possible long-term movements. We adopt the same idea and construct a cost map, $M_{route}$ using a route planner [64], where the cost map contains possible routes that the OV will take (6). Also, we recognize that a vehicle in the parking lot normally runs in two modes, “maneuvering” and “cruising”. Vehicles in maneuvering mode change the steering frequently and deviate from the routes (black dashed
line in Fig. 4) in the cost map in order to park or leave the narrow parking spot. Vehicles in cruising mode have small or steady steering angles and generally follow one of the routes. Vehicles are in this mode when they first enter the parking lot and are approaching a parking spot or when they got out of the parking spot and are leaving the parking lot. Including the route information, an OV that has n routes to follow will have 2n possible modes (Fig. 4). To determine the mode m at time step k, i.e., m_k. Bayesian framework is employed to keep track of the belief of each mode, i.e., b(m_k). The process is described in Alg. 1, lines 5~8. We perform the prior belief update p(m_k) = T_k(b(m_{k-1}) based on the previous belief, b(m_{k-1}). The posterior p(m_k|X_{cc,k},X_{H,k}) is proportional to the prior multiplied with the conditional probability of the motion estimation and prediction given the mode, i.e., p(m_k)p(X_{cc,k},X_{H,k}|m_k). The Boltzmann policy is one common way to design this conditional probability [65], i.e., p(X_{cc,k},X_{H,k}|m_k) \propto \exp(-\rho \cdot M_{route}(m_k,X_{cc,k},X_{H,k}) f(m_k,X_{cc,k}). The function M_{route}(m_k,X_{cc,k},X_{H,k}) compares the OV states and predictions with the routes:

\[
M_{route}(m_k,X_{cc,k},X_{H,k}) = \min_{i} \|X_{m,k,i} - X_{cc,k}\|_W^2 + \sum_{h} \min_{i} \|X_{m,k,i} - X_{h,k}\|_W^2,
\]

where X_{m,k,i} is the i\textsuperscript{th} waypoint of the route in mode m_k = m, m \in \{1,...,2n\}, \|v\|_W^2 = v^\top W v, and W_1 and W_2 are weighting matrices. The function f(m_k,X_{cc,k}) is proportional to the magnitude of the OV’s steering angle and the deviation of the OV’s heading angle from the final heading angle of the route. Finally, we normalize p(m_k|X_{cc,k},X_{H,k}) to obtain b(m_k) and take the value of m_k with the largest belief to be \hat{m}_k.

4) Safety margin and safety bound: When the OV is in cruising mode, the predictor calculates the safety margins s_{H,k} = [s_{k+1},...,s_{k+H}]\top (red areas in Fig. 5) according to Alg. 1, lines 9 and 10. The safety margin of the hth future time step is an ellipsoid and the length of the principal semi-axes s_{k+1} (h = 1, ..., H) are proportional to the differential entropy of the mode belief and the covariance from the motion estimation, i.e., s_{k+1,h} \propto (b_{uncertainty} C^T P_{m_{k+1}} C). Since the movements of the OV in maneuver mode (colored in blue in Fig. 4) are hard to predict, the predictor generates a safety bound (the bound of a convex hall of the OV’s pose history, orange lines in Fig. 5) so that the planner behaves more conservatively and keeps the ego AV away from the hardly predictable OVs. Notice that the safety margin and safety bound can also be applied to other moving obstacles such as pedestrians or motorbikes given their kinematic models and routes information.

D. Strategic Motion Planner

With the reference trajectory P_{ref}, the strategic motion planner runs the main module, the MPC-based safety controller, and two supporting modules: the retreat planner and the repair planner (Fig. 2). Both planners are activated if the ego AV’s current location and reference trajectory is invalidated by the OV’s movements, respectively. The strategic motion planner is summarized in Alg. 2.

1) MPC-based safety controller: The safety controller tracks the reference trajectory P_{ref} given the safety margin and the safety bound. With P_{ref}, we use an optimization-based planner to compute tracking motions in an MPC framework. Let X_{ref,k} be the segment of P_{ref} to track at time step k. X_{ref,k} is selected and trimmed so that it will not violate the safety margin (in all modes) nor the safety bound (in “maneuver” modes). The trajectory tracking problem is formulated as follows:

**Problem 3.2:** Given the planning horizon H, the vehicle model (2), and the reference segment X_{ref,k}, the optimization planner solves the problem

\[
u_k = \arg \min_{u_k} \| F(X_k) + G(X_k, u_k) - X_{ref,k} \|_{W_1}^2,
\]

s.t. \( X_k = F(X_k) + G(X_k, u_k), \)
\( (X_k, u_k) \in \Gamma_k, \)

where X_k = [X^\top_{k+1}, X^\top_{k+2},...,X^\top_{k+H}]^\top, u_k = [u_k^\top, u_{k+2}^\top, u_{k+h+1}^\top] \hfill (7)

\[ u_k \in [-u_{max}, u_{max}], h = 1, ..., H, \]

Problem 3.2 can be readily formulated as a non-convex optimization problem using various software tools, e.g., CasADi [66], and solved using nonlinear programming solvers, e.g., IPOPT. With the reference path serving as a warm start, the average solving time is around 0.06 second. Details are omitted here.
2) Retreat planner: The retreat planner deals with scenarios when stopping or staying on the original reference is deemed unsafe. This can happen when the OV drives toward the ego AV, and its motion largely differs from the previous prediction - possibly violating the safety margin and causing a safety threat. Therefore, the ego AV needs to find a path and retreat from the emergency. The retreat planning movement is not a standard navigation problem because the ego AV hasn’t had a safe goal. Instead, it needs to explore the environment to find the best goal, and thus we propose a search-based retreat planner which explores the space and quickly finds a retreating trajectory.

As a variant of A*-based algorithms, the retreat planner constructs a tree \( T = (\mathcal{V}, \mathcal{E}) \) composed of a node set \( \mathcal{V} \subset C_{\text{free}} \) and an edge set \( \mathcal{E} \), where \( E(X_i, X_j) \in \mathcal{E} \) represents a feasible short path between \( X_i \) and \( X_j \), \( C_{\text{free}} \) is implicitly obtained by checking collisions with obstacles in the parking lot map \( M_{\text{map}} \). Let \( \mathcal{M} \) denote a finite set of motion primitives pre-computed through available control actions, and \( V_{\text{max}} \) denotes the maximum number of nodes allowed. The retreat planner constructs a tree \( T \) from \( X_k \) (the configuration when the retreat planning starts) and expands it according to a cost function \( F(\cdot) \) which sums up the heuristic value \( h(\cdot) \) and the arrival cost \( g(\cdot) \). The heuristic (8) is calculated based on a collision field as shown in Fig. 6. The field is a weighted sum of Gaussian distributions centered at waypoints of both the predicted trajectory \( X_{H,k} \), i.e., \( X_{h,k} \), and the routes on the cost map, i.e., \( X_{m_k,i} \), and

\[
h(X) = \sum_{m_k,i} b(m_k)e^{-\|X-X_{m_k,i}\|^2/2\sigma_4} + \sum_{h} c e^{-\|X-X_{h,k}\|^2/2\sigma_5}, \quad (8)
\]

where \( c \) is a weighting constant. The planning is completed if the ego AV finds a trajectory that keeps it away from the OV at a safe clearance. If the number of nodes in \( T \) reaches \( V_{\text{max}} \), the trajectory giving the maximum clearance is chosen.

3) Trajectory repair planner: OV’s movements could invalidate the ego AV’s reference trajectory. Fig. 7 illustrates one such case, where an OV, represented by the blue box, stops on the AV’s reference trajectory, represented as the light-blue line. The safety controller will command the ego AV to stop on the reference trajectory when the area in front is infeasible. Unless receiving a new reference trajectory, the safety controller will stop the ego AV and wait for the OV to clear - not efficient if the OV stops for a long time. It is reasonable to update \( P_{\text{ref}} \), so that the safety planner can command the ego AV to go around the OV and merge back to the original path. We notice that the repaired trajectory usually lies in the same homotopy class as the original one, which makes an optimization-based repairing strategy a viable solution. To obtain a repaired path quickly, we conduct repair planning over 2D space, i.e., \( X_{\text{repair}} = [x, y]^T \), and modify the constraints accordingly. It is understood that the resultant path, despite being collision-free, cannot always be followed accurately, causing the AV to collide into obstacles. We therefore verify the path and accept the repaired trajectory (as shown in Fig. 7) only if it passes the kinematic feasibility check. If the repairing fails, the central control will be notified to take over the repairing task.

IV. Empirical Evaluation

The proposed system is tested by simulation, which is conducted on a 6-core Intel i7 3.7GHz desktop with Matlab R2020a. The prediction horizon is 10, i.e., \( H = 10 \). The integrated system is set to runs at a rate of 4 Hz (all calculation in each time step is finished within 0.1 second). Due to space limitation, this section presents one of the simulation results as an example. More simulation results can be found in the video available at jessicaleu24.github.io/ICRA2022.html.

As shown in Fig. 8(a), the ego AV (red box) first performs parking by tracking the reference trajectory \( P_{\text{ref}} \) (light-blue line) while avoiding collision with the OV (blue box). The safety margins are illustrated by red shaded areas, where the one with black edge is for the current time step and the one without edge is for the \( H \)th future time step (the value is
As the OV comes out from its parking spot, it moves towards the ego AV. The ego AV needs to retreat temporarily to make space for the OV (Fig. 8(b)~(d)). Although the ego AV might be able to avoid collision by backing up along the original reference, this movement may be dangerous since that is the direction where the OV is heading towards and will potentially block the OV. This showcases the importance of considering the long-term mode to construct a collision field (as shown in Fig. 6) of the OV in retreat planning. In Fig. 8(b)-8(c), the retreat planner is activated to update the $P_{ref}$. When violation happens again (Fig. 8(d)), the retreat planner updates $P_{ref}$ once more. As the OV completely leaves the parking spot, the estimator detects the mode switch (from “maneuver left” to “cruise (exit) left” in Fig. 9), therefore the safety bound is removed (Fig. 8(e)) and the safety controller continues to follow the reference path (Fig. 8(e)-8(f)). Later, the OV stops at the entrance of the road and blocks the original reference trajectory of the ego AV. Therefore, the ego AV tries repairing the trajectory (Fig. 8(g)). However, the repairing fails (the solver converges to an infeasible red path). Because of the narrow space, cusps are required in the maneuver, which is generally hard for an optimization-based planner to generate. Therefore, a new trajectory from the central control is required. Once the updated trajectory $P_{ref}$ (Fig. 8(h)) is received, the safety controller will start following it until the ego AV reaches the goal. In the video, we show a successful path repair in demo 1 and the effectiveness of the safety bound (so that the retreat planner won’t be triggered unnecessarily) in demo 3.

V. CONCLUSION AND FUTURE WORK

This paper presents an integrated motion planning strategy for an AV to park in dynamic environments. A hybrid environment predictor incorporates the model-based short-term motion prediction and a driver behavior cost-map to make long-term prediction of an OV. A strategic motion planner, composed of an MPC-based safety controller, a search-based retreat planning, and an optimization-based repairing planner, strikes a good balance between safety, plan feasibility, and smooth maneuver by leveraging the advantages of optimization-based and search-based ideas. Depending on the predictor and the AV’s objective, the strategic motion planner generates safe and smooth trajectories that bring the AV to the target directly or through an intermediate safe spot to yield to OVs. Simulation results demonstrate that the proposed approach enables the ego AV to plan safely and move smoothly in complicated dynamic parking environments. Future work includes: 1) perform rigorous analysis of the integration strategy, e.g., formal method analysis; 2) generalize and verify the proposed strategy in multi-OVs environment; 3) perform real-time simulation in platforms, for instance dSPACE, for comprehensive assessment of performance and computation load.

VI. ACKNOWLEDGEMENT

The authors would like to thank Karl Berntop, Rien Quirynen, Marcel Menner, and Abraham P. Vinod for their valuable comments.
REFERENCES


