Trajectory Optimization for Autonomous Overtaking with Visibility Maximization

Hans Andersen[1], Wilko Schwarting[2], Felix Naser[3], You Hong Eng[3], Marcelo H. Ang Jr.[1], Daniela Rus[2], and Javier Alonso-Mora[4]

***Abstract*— In this paper we present a trajectory generation method for autonomous overtaking of static obstacles in a dynamic urban environment. In these settings, blind spots can arise from perception limitations. For example, the autonomous car may have to move slightly into the opposite lane in order to cleanly see in front of a car ahead. Once it has gathered enough information about the road ahead, then the autonomous car can safely overtake. We generate safe trajectories by solving, in real-time, a non-linear constrained optimization, formulated as a Receding Horizon planner. The planner is guided by a high-level state machine, which determines when the overtake maneuver should begin. Our main contribution is a method that can maximize visibility, prioritizes safety and respects the boundaries of the road while executing the maneuver. We present experimental results in simulation with data collected during real driving.**

Fig. 1. Unexpected static obstacle in the form of an illegally parked truck on a two-way street

I. Introduction

Autonomous vehicles are anticipated to ease road congestion and reduce the number of traffic accidents, by eliminating human error. However, autonomous driving in urban environments poses different challenges compared to highway driving, as the environment is less structured and predictable, and there are different traffic rules and human driving characteristics that are unique in each urban area. Therefore, decision making for autonomous driving in urban areas is a particularly interesting research area that has seen a lot of interest in recent years. Recent surveys by Katrakazas et al. [1], Paden et al. [2] and Pendleton et al. [3] describe different approaches with their strengths and limitations.

Reacting to potentially hazardous situations is one of the key issues in autonomous driving in urban environment. Occluded objects may come out of the autonomous vehicles' blind spot unpredictably, and therefore, the vehicle may not have enough time to plan for appropriate action in due time.

A few example cases have been available in the literature such as the study by Althoff et al. [4] considers uncertainties from measurement of other traffic participants stochastically in order to assess the safety of planned trajectories. Hayashi et al. [5] proposed a method to evade obstacles that suddenly appear from the side of the road by computing a geometrically optimized path, and executing the maneuver by steering and brake actuation.

One particular scenario that we encounter very frequently during our autonomous vehicle deployment at the One-North area in Singapore is depicted in Figure 1. In this scenario, a truck is illegally parked on the vehicle's ego lane, and therefore has to be overtaken. However, as this is a two-way traffic, the overtaking implies that the vehicle invades to the opposite lane, and therefore will take the traffic head-on, causing a safety hazard.

Guo et al. [6] proposed a solution to circumventing the illegally parked vehicle by finding a lead vehicle in the ego lane and follow its behavior to generate a trajectory that is based on a cubic spline model with mass-spring damper system. However, this approach may fail if there are no leading vehicles in the ego lane or if the intention of the vehicle is unknown, as the urban traffic rules can be complicated and very dynamic.

We have observed the following behavior of human drivers facing the described scenario: they will first decelerate the vehicle, and move closer to the center of the lane and asses the traffic on the opposite lane as well as the distance that the driver has to overtake, before finally overtaking the obstacle and merging back to the ego lane.

In this paper, we consider the problem of autonomous overtaking of unexpected obstacles on a two-way street in an urban environment. The main contributions of this paper are threefold.

- A Receding Horizon (or Model Predictive Control) formulation that maximizes the amount of information

[1]Hans Andersen and Marcelo H. Ang Jr. are with the National University of Singapore, Singapore hans.andersen@u.nus.edu, mpeangh@nus.edu.sg

[2]Wilko Schwarting and Daniela Rus are with the Massachusetts Institute of Technology, Cambridge, MA, USA {wilkos, rus}@csail.mit.edu

[3]You Hong Eng and Felix Naser are with the Singapore-MIT Alliance for Research of Technology, Singapore {youhong, felix.naser}@smart.mit.edu

[4]Javier Alonso Mora is with the Delft University of Technology, the Netherlands J.AlonsoMora@tudelft.nl

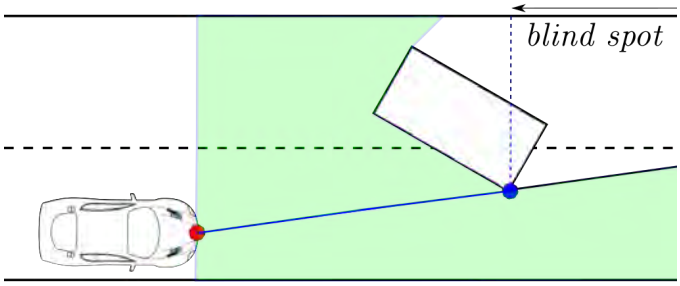

Fig. 2. Blind spot caused by an occluding obstacle in the vehicle's ego lane. The green shaded region shows the area visible by the sensor and the area that is not covered by the sensor is shown in white.

that the autonomous vehicle gains along its trajectory. This allows the vehicle to make a more informed and therefore safer decision before overtaking the obstacle.
- A higher-level decision making system for behavior planning that performs safety checks, and ensures that the vehicle is able to safely overtake the obstacles of unknown length, and safely return to its lane, whenever necessary.
- Simulation results that demonstrate the capabilities of the algorithm in generating safe and optimal trajectories for autonomous overtaking in urban environment.

Our method can be briefly summarized as follows. Assuming that the lane boundaries are known a priori, the perception system will detect obstacles in both the ego lane and the opposite lane. The perception system also estimates the amount of blind spot caused by the occluding obstacle in the ego lane (Fig. 2). The state machine then makes a decision based on inputs from the perception system, that determines the behavior of the system. Different system behaviors - such as overtaking or remaining on the lane - modify the tunable weights, but not the formulation of the Receding Horizon planner. The autonomous vehicle then generates a safe trajectory by solving a non-linear constrained optimization in a MPC style. The cost and reward terms of the problem consist of: path following errors, progress along the desired path, velocity error, size of blind spot (visible area), and inputs. The constraints of the problem are the motion model of the vehicle, the state and input bounds, collision avoidance with respect to obstacles, maximum yaw rate, angular deviation from the path, and maintenance within the road boundaries. An off-the-shelve non-linear optimizer is then periodically called to solve the optimization problem, and the optimal input is given to the system.

The remaining of this paper is organized as follows. Related works are reviewed in Section II. Preliminaries are introduced in Section III. The trajectory planner method is then described in Section IV, followed by the high-level state machine in Section V. The simulation set up and results, including supporting perception modules are discussed in Section VI. Section VII concludes this paper.

II. Related Works

A practical application of motion planning is autonomous overtaking, which has also been widely researched in the literature. Trajectory generation for static overtaking [7], first real road applications [8] and the use of fuzzy controllers that mimic human behavior and reactions during overtaking maneuvers are examples of ongoing research. Other recent examples include the work by Cunningham et al. [9] that considers multiple policies and a user defined cost function to determine when to overtake or stay in the ego lane. Schlechtriemen et al. [10] proposed a method to abstract dynamic objects and static obstacles as time dependent geometric bodies, and plan the trajectory based on the abstraction.

Sampling-based methods, such as RRT and its variants [11], are popular for trajectory planning. Their strength is probabilistic completeness. However, probabilistic motion planning suffers from inherent accuracy due to discretization limits, and the computational complexity that rises exponentially as the dimensionality of the planning state space increases. Autonomous driving in urban environment often requires smoothness and accurate modeling the car dynamics. These requirements render Receding Horizon optimization schemes, such as MPC, well suited for solving these problems.

MPC has seen many applications in autonomous driving, specially in trajectory tracking applications. Wang et al. [12] proposed a conflict-probability-estimation-based overtaking for intelligent vehicles using MPC. A recent example from Cairano et al. [13] guarantees that the controller tracks piecewise-clothoidal trajectories within a preassigned lateral error bound. It has also been applied to collision avoidance at intersections [14], overtaking [15], obstacle avoidance for critical maneuvers [16], and as a parallel autonomy planner, in which the autonomous system works hand in hand with a human driver [17]. Ziegler et al. [18] have proposed an MPC based trajectory planner for autonomous driving along the Bertha- Benz Memorial Route. Static and dynamic obstacles are represented as polygons, and road boundaries are used as heuristics on which side of the obstacle to overtake from.

In this paper, we also employ a MPC-based local motion planner, which captures the dynamics of the car and avoids obstacles. In contrast to previous works, we also consider visibility maximization, to generate overtaking trajectories that take into account the perception limitations of the ego vehicle.

III. Preliminaries

A. Vehicle Model

Denote by t_0 the initial planning time and by Δt_i the i-th timestep of the planner. We consider a discrete time system with time $t_k = t_o + \sum_{i=1}^{k} \Delta t_i$. The configuration of the ego vehicle at time k is denoted as $\mathbf{z}_k = [\mathbf{p}_k, \phi_k, \delta_k, v_k] \in \mathcal{Z}$, where $\mathbf{p}_k = [x_k, y_k]$ is the position, ϕ_k is the orientation, v_k is the linear velocity, and δ_k is the steering angle of the vehicle in the global frame. The control input to the system

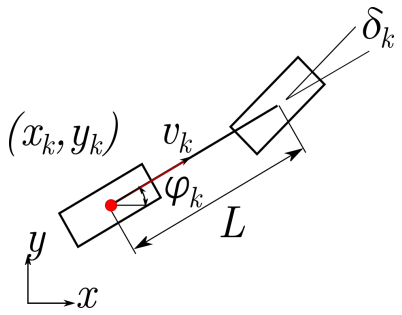

Fig. 3. Kinematic bicycle model of an Ackermann-steered vehicle.

at time k is denoted as $\mathbf{u}_k = [u_k^\delta, u_k^a] \in \mathcal{U}$, where u_k^δ is the steering rate $\dot{\delta}_k$ and u_k^a is the linear acceleration a_k.

In this work we employ a bicycle kinematic model Fig. 3, which approximates a four wheeled Ackermann steered car. This model is often used in literature to derive steering control laws as it approximates the motion of the vehicle reasonably well at low speeds and moderate steering angles, which is the common driving condition in urban environment. The method is general and other vehicle models can be considered.

The continuous state-space equation of the system can be written as

$$\underbrace{\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\phi}\\ \dot{\delta}\\ \dot{v}\end{bmatrix}}_{\dot{\mathbf{z}}} = \begin{bmatrix} v\ cos(\phi) \\ v\ sin(\phi) \\ (\frac{v\ tan\delta}{L}) \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\\ 0 & 0\\ 1 & 0\\ 0 & 1\end{bmatrix} \underbrace{\begin{bmatrix}u^\delta\\ u^a\end{bmatrix}}_{\mathbf{u}} \tag{1}$$

where L is the wheelbase of the vehicle. The discrete time state space system

$$\mathbf{z}_{k+1} = f(\mathbf{z}_k, \mathbf{u}_k) \tag{2}$$

can be approximated with the integration model $\mathbf{z}_{k+1} = f(\mathbf{z}_k, \mathbf{u}_k) = \mathbf{z}_k + \int_k^{k+\Delta t} \dot{\mathbf{z}}\ \mathrm{d}t$. The fourth order Runge-Kutta integration method is used in the optimizer for sufficient accuracy.

The states of the system (steering angle $\|\delta\| \leq \delta_{max}$, longitudinal speed $v \leq v_{max}$ yaw rate $\|\dot{\phi}\| \leq \dot{\phi}_{max}$), as well as the control inputs (steering rate $\|u^\delta\| \leq \dot{\delta}_{max}$ and acceleration $a_{min} \leq u^a \leq a_{max}$) are limited to our vehicle's specifications. These constraints are added in the Receding Horizon planner.

B. Path Representation and Tracking

In nominal conditions the autonomous car follows centre line of the driving lane. To track the centre line while avoiding obstacles we follow [17] and formulate a Model Predictive Contouring Control (MPCC) [19] problem. MPCC optimizes the progress along the path, while considering nonlinear projection of the vehicle's position onto the desired path. The desired path contour is the centre line of the lane, and is parametrized as piecewise continuous, continuously differentiable cubic splines with multiple knots along the path.

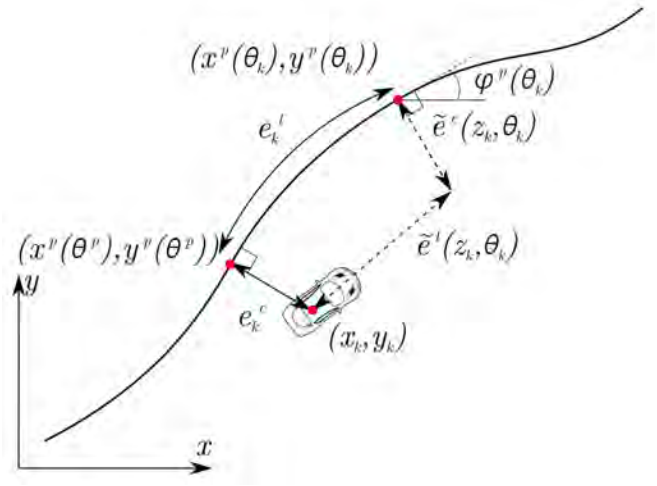

Fig. 4. Approximation of lag and contouring cost along the path.

At a given time k, the vehicle's anchor point position $\mathbf{p}_k$ tracks a continuously differentiable reference path $(x^p(\theta), y^p(\theta))$ with path parameter θ. The tangential and normal vectors to the path are given by

$$\mathbf{t}(\theta) = \begin{bmatrix}\frac{\partial x^p(\theta)}{\partial \theta}\\ \frac{\partial y^p(\theta)}{\partial \theta}\end{bmatrix},\ \mathbf{n}(\theta) = \begin{bmatrix}\frac{-\partial y^p(\theta)}{\partial \theta}\\ \frac{\partial x^p(\theta)}{\partial \theta}\end{bmatrix}, \tag{3}$$

and the heading of the path is given by

$$\phi^p(\theta) = arctan\left(\frac{\frac{\partial x^p(\theta)}{\partial \theta}}{\frac{\partial y^p(\theta)}{\partial \theta}}\right). \tag{4}$$

The vehicle's progress along the path is parametrized by arc length s with $(\partial\theta/\partial s = 1)$, and can be approximated for a small step by integrating the velocity of the vehicle over time $s = \int v\ \mathrm{d}t$. Assuming that the vehicle tracks the given path with sufficient accuracy, we can approximate the change in path parameter θ by

$$\Delta\theta \approx \Delta s = v\Delta t, \tag{5}$$

and therefore for one timestep, the evolution of parameter can be approximated by

$$\theta_{k+1} = \theta_k + v_k \Delta t_k, \tag{6}$$

where $v_k \Delta t_k$ is the approximated progress along the path at time state k. In the ideal case, the path parameter $\theta^p(x_k, y_k)$ should be computed in closed form inside the optimizer as the projection of (x_k, y_k) to the path. However, this process involves computing the computationally expensive optimization routine

$$\theta^p(x_k, y_k) = \underset{\theta'_k}{\operatorname{argmin}}\ (x_k - x^p(\theta'_k))^2 + (y_k - y^p(\theta'_k))^2. \tag{7}$$

For computational efficiency, $\theta^p(x_k, y_k)$ is approximated by the evolution over time during the optimization process.

Approximating $\theta^p(x_k, y_k)$ with θ_k introduces two errors if the vehicle's position deviates from the desired reference

path, namely the longitudinal (lag) error e_k^l along the path and the lateral (contouring) error e_k^cnormal to the path as shown in Fig. 4.

The lag error can be approximated by projecting the position error of the vehicle's position to θ_k along the path's tangent vector $\mathbf{t}(\theta_k)$, formally

$$\begin{aligned}\tilde{e}^l(\mathbf{z}_k, , \theta_k) &= \frac{\mathbf{t}(\theta_k)^T}{\|\mathbf{t}(\theta_k)\|}\begin{bmatrix} x_k - x^p(\theta_k) \\ y_k - y^p(\theta_k)\end{bmatrix} \\ &= -\cos\phi^p(\theta_k)(x_k - x^p(\theta_k)) \\ &\quad -\sin\phi^p(\theta_k)(y_k - y^p(\theta_k))\end{aligned} \tag{8}$$

The contouring error, which measures how far the vehicle deviates from the reference path, can be approximated by projecting the position error of the vehicle's position to θ_k along the path's normal vector $\mathbf{n}(\theta_k)$, formally

$$\begin{aligned}\tilde{e}^c(\mathbf{z}_k, , \theta_k) &= \frac{\mathbf{n}(\theta_k)^T}{\|\mathbf{n}(\theta_k)\|}\begin{bmatrix} x_k - x^p(\theta_k) \\ y_k - y^p(\theta_k)\end{bmatrix} \\ &= \sin\phi^p(\theta_k)(x_k - x^p(\theta_k)) \\ &\quad -\cos\phi^p(\theta_k)(y_k - y^p(\theta_k))\end{aligned} \tag{9}$$

The errors are formulated into a Receding Horizon planner (described in the forthcoming Section IV) as additional cost terms, while the progress along the path is formulated as reward,

$$J_{MPCC}(\mathbf{z}_k, \theta_k) = \mathbf{e}_k^T Q \mathbf{e}_k - \gamma v_k \Delta t_k \cos(\phi_k - \phi^p(\theta_k)), \tag{10}$$

where $Q \in \mathbb{S}_+^2$ and $\gamma \in \mathbb{R}_+$ are predefined weights and the path error vector $\mathbf{e}_k$ is given by the approximated lag and contouring errors,

$$\mathbf{e}_k = \begin{bmatrix}\tilde{e}^l(\mathbf{z}_k, \theta_k) \\ \tilde{e}^c(\mathbf{z}_k, \theta_k).\end{bmatrix} \tag{11}$$

IV. Trajectory Generation

A. Road Boundaries and Obstacle Representation

We follow the description of road boundaries and obstacles by [17] and modify it to account for rectangular obstacles. The ego vehicle is represented as a union of a set of 4 circles $\mathcal{R}^j(\mathbf{z}_k, j \in \{1, \cdots, 4\})$ of radius r_{disc}, which is chosen in a conservative manner, as shown in Fig. 5 to enclose the vehicle's footprint.

The lateral distance $d(\mathbf{z}_k, \theta_k)$ of the ego vehicle's position to the reference path is given by the normal projection vector at θ^P. Since we approximate $\theta^P \approx \theta_k$, we can approximate the lateral distance by the contouring error $d(\mathbf{z}_k, \theta_k) \approx \tilde{e}^c(\mathbf{z}_k, \theta_k)$.

The drivable region at θ_k is limited by the road boundaries. The left road boundary is at distance b_l and the right road boundary is at distance b_r. To ensure that the ego vehicle drives within the limits of the road, we enforce the constraint

$$b_l(\theta_k) + w_{max} \leq d(\mathbf{z}_k, \theta_k) \leq b_r(\theta_k) - w_{max}, \tag{12}$$

where w_{max} is an upper bound of the vehicle's outline projected onto the reference path's normal. For practical purposes, it can be set as an additional padding to r_{disc}. To ensure that the vehicle can follow the desired path well in situations where road boundaries are tight, we introduce an additional constraint on the path heading difference,

$$\|\phi_k - \phi^p(\theta_k)\| \leq \Delta\phi_{max}. \tag{13}$$

Each unexpected obstacle i, such as the wrongly parked truck, is modeled by a rectangle of length a_{obs}^i and width b_{obs}^i, such that $a_{obs}^i \geq b_{obs}^i$, whose centroid is located at (x_{obs}^i, y_{obs}^i) in the global reference frame and which has orientation ϕ_{obs}^i.

A coordinate frame is attached to each obstacle, originating at its centroid, with the x axis parallel to its length and the y axis parallel to its width. Consider the origin of the *j*-th circle that describes the footprint of the vehicle, its position in the *i*-th obstacle's coordinate frame is $x_{disc(j)}^{obs(i)}, y_{disc(j)}^{obs(i)}$. The collision constraint between the obstacle and the circle at time k can be formulated as

$$\Delta x_{k,disc(j)}^{obs(i)} = \max(\frac{-a_{obs}^i}{2}, \min(x_{k,disc(j)}^{obs(i)}, \frac{a_{obs}^i}{2})) \tag{14a}$$

$$\Delta y_{k,disc(j)}^{obs(i)} = \max(\frac{-b_{obs}^i}{2}, \min(y_{k,disc(j)}^{obs(i)}, \frac{b_{obs}^i}{2})) \tag{14b}$$

$$c_{k,disc(j)}^{obs(i)}(\mathbf{z}_k) = \frac{(\Delta x_{k,disc(j)}^{obs(i)})^2 + (\Delta y_{k,disc(j)}^{obs(i)})^2}{(r_{disc} + r_{overtake})^2} \geq 1, \tag{14c}$$

where $r_{overtake}$ is the additional safe overtaking distance.

We also employ a dynamic virtual bumper (DVB) [20] to generate a safe advisory speed v_{ref} for the vehicle. The DVB is a tube-shaped zone with its centerline as the vehicle's local path, and its width and height given by quadratic functions dependent on the vehicle's speed v_k, and the obstacles in the region.

The speed deviation is incorporated into the optimization as an additional cost term

$$J_v(\mathbf{z}_k, \theta_k) = \zeta(v_{ref} - v_k)^2, \tag{15}$$

where $\zeta \in \mathbb{R}_+$.

B. Visibility Maximization

Consider an autonomous vehicle equipped with a sensor (we use a LIDAR, but the method can be applied to other sensor types) with a limited sensing range and field of view. Now consider a scenario, when the vehicle is approaching an obstacle in a straight line section, such as the case shown in Fig. 6. We assume that the car is driving on the left side of the road and therefore has to overtake on the right side of the obstacle. Our goal is generate a motion for the vehicle such that the visibility ahead of the obstacle is maximized. For this, we first provide a definition of blind spot and then describe the cost term to be added in the Receding Horizon planner.

In the top image of Fig. 6 we show a case where an obstacle in the ego lane generates a complete occlusion along the left road boundary. For obstacle 1, we consider a frontier point (blue dot) located to the right of the centreline of the sensor located at (x_{sensor}, y_{sensor}) and with orientation (ϕ_{sensor}). We define the frontier point of a set of

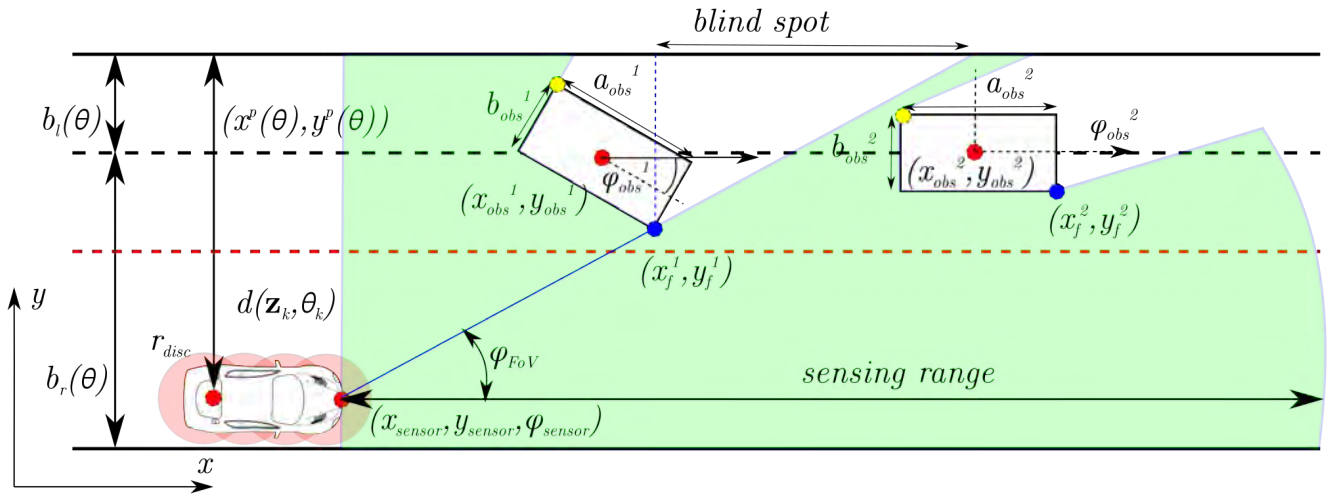

Fig. 5. Figure illustrating a maneuver to increase the visibility, i.e. the area seeing by the autonomous car that was previously occluded. Road boundaries, obstacle, and blind spot representation of an overtaking vehicle. Solid black lines represent road boundaries, a dashed black line represents the centre line of the ego lane, dashed red lines represent the road boundary of the ego lane. The green shaded region shows the area visible by the sensor and the area that is not covered by the sensor is shown in white.

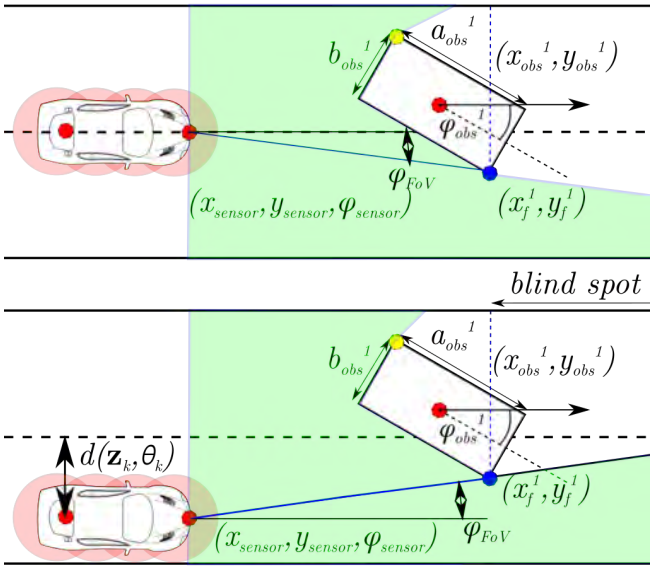

Fig. 6. Complete occlusion (top) and blind spot (bottom) of a vehicle approaching an obstacle.

measurement points as the point, at position (x_f^1, y_f^1), that has the smallest field-of-view angle ϕ_{FoV}, measured from the centreline of the sensor. When complete occlusion happens, ϕ_{FoV} has negative value. The yellow point in the figure is the exact opposite of the frontier point, i.e. it has the largest field-of-view angle in the set.

As the vehicle moves towards the right boundary of the ego lane, as shown in Fig. 5 (bottom), (ϕ_{FoV}) increases. When the angle (ϕ_{FoV}) is positive, a blind spot appears, i.e. a section along the left road boundary is not covered by the sensor. This event causes a loss of information, which is necessary to determine if the overtake maneuver is safe and the autonomous vehicle can safely return to its ego lane. As seen in Fig. 5, the autonomous vehicle may need to move into the opposite lane to minimize the blind spot, and therefore maximize visibility, until it is able to make a safe decision to overtake.

Therefore, to minimize the blind spot along a trajectory we have to maximize the visibility, i.e. the field-of-view. The angle ϕ_{FoV} can be encoded into the optimization problem as a reward term

$$J_{FoV}(\mathbf{z}_k, \theta_k) = -\lambda \phi_{FoV}, \tag{16}$$

where $\lambda \in \mathbb{R}_+$.

C. MPC Formulation

We formulate a Receding Horizon planner that includes all of the aforementioned cost terms and constraints. We write the following non-linear constrained optimization,

$$\underset{u_{0:N-1}}{\text{minimize}} \quad J_t(\mathbf{z}_N, \theta_N) + \sum_{k=0}^{N-1} J(\mathbf{z}_k, \mathbf{u}_k, \theta_k)\Delta t_k \tag{17a}$$

$$\text{subject to} \quad \mathbf{z}_{k+1} = f(\mathbf{z}_k, \mathbf{u}_k) \tag{17b}$$

$$\theta_{k+1} = \theta_k + v_k \Delta t_k \tag{17c}$$

$$\mathbf{z}_{min} \leq \mathbf{z}_k \leq \mathbf{z}_{max} \tag{17d}$$

$$\mathbf{u}_{min} \leq u_k \leq \mathbf{u}_{max} \tag{17e}$$

$$\|\dot{\phi}\| \leq \dot{\phi}_{max} \tag{17f}$$

$$\|\phi_k - \phi^p(\theta_k)\| \leq \Delta\phi_{max} \tag{17g}$$

$$b_l(\theta_k) + w_{max} \leq d(\mathbf{z}_k, \theta_k) \leq b_r(\theta_k) - w_{max} \tag{17h}$$

$$c_{k,disc(j)}^{obs(i)}(\mathbf{z}_k) > 1, i = \{1, \cdots, m\}, \quad j = \{1, \cdots, 4\} \tag{17i}$$

$$\forall k \in \{0, \cdots, N\},$$

where N is the prediction horizon and m is the number of detected obstacles.

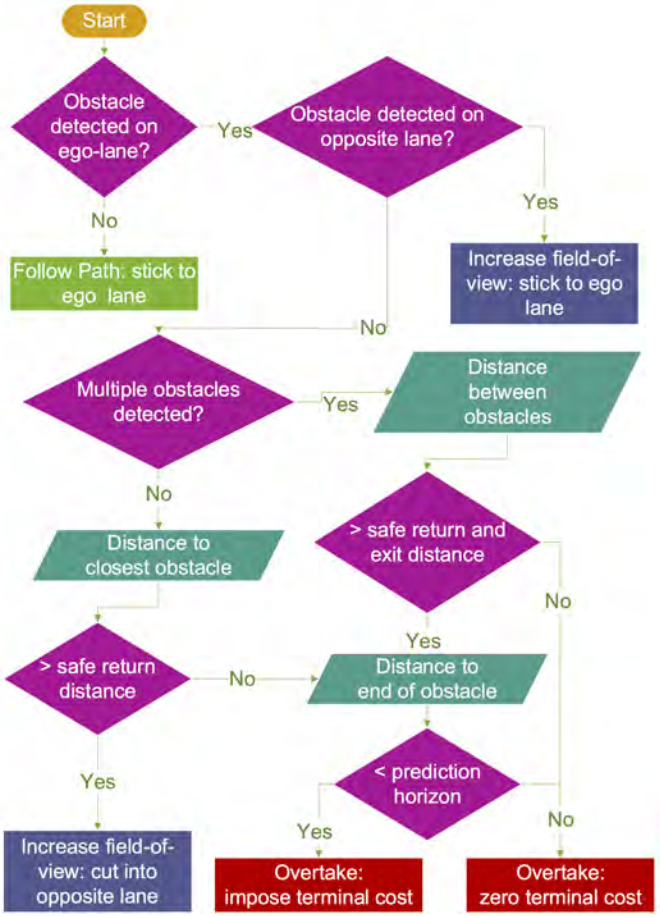

Fig. 7. Behaviour planning conditional flowchart.

Recalling the previous sections, the cost term is given by

$$J(\mathbf{z}_k, \mathbf{u}_k, \theta_k) = J_{MPCC}(\mathbf{z}_k, \theta_k) + J_v(\mathbf{z}_k, \theta_k) + J_{FoV}(\mathbf{z}_k, \theta_k) + \mathbf{u}_k^T R \mathbf{u}_k, \quad (18)$$

where $R \in \mathbb{S}_+^2$ is the control input cost.

The terminal cost is defined as

$$J_t(\mathbf{z}_N, \theta_N) = \mathbf{e}_N^T Q_t \mathbf{e}_N, \quad (19)$$

where $Q_t \in \mathbb{S}_+^2$ is a design parameter. This additional terminal cost is added when the vehicle is about to finish its overtaking sequence and merge back into the ego lane, as we discuss in the next section.

V. Behavior Planning

A conditional state machine, shown in Fig. 7, has been designed for higher-level behavior planning. The behavior layer outputs the value for the design parameters of the trajectory planner. Every decision made by the behavior planner is fed into the same MPC formulation described in Section IV, i.e. it only modifies the parameters of the optimization problem. We assume that the perception system is able to give a Boolean signal on the clearance of the opposite lane, including the prediction of incoming traffic.

The first case is nominal driving. When there are no obstacles detected on the ego lane, the vehicle should follow the centre of the ego lane as closely as possible, with high penalty for lag error, contouring error, and speed deviation.

The second case is maximizing visibility without invading the opposite lane. If there are obstacles detected on both the ego lane and the opposite lane, the vehicle should move towards the centre of the lane, without leaving the ego lane, to both maximize visibility, as well as to anticipate the instance when the opposite lane becomes available. Furthermore, when the autonomous vehicle is in the opposite lane, and an obstacle is detected for the first time, the vehicle should return to its ego lane while still maximizing visibility. This is enabled by giving large reward to the field of view term, as well as reducing the lag and contouring cost, and limiting the road boundaries to the ego lane.

The third case is maximizing visibility while invading the opposite lane. If there are no obstacles detected in the opposite lane, the vehicle is then able to cross to the opposite lane to further maximize visibility. We further ensure that the vehicle is able to safely return to its driving lane if an incoming obstacle is detected in the opposite lane, i.e. the vehicle should not use the width of the opposite lane in full when it is still not sure that it will execute the overtake maneuver. We guarantee this by checking the closest distance to the obstacle. If this distance is above a conservative number of $6 \times \frac{tan(\delta_{max})}{L}$, or six times the minimum turning radius of the vehicle, it should only use half of the opposite lane. When the vehicle is fully committed to the overtaking maneuver, the road boundary can then be loosened further to the full width of the opposite lane. A possible extension to this heuristic is a full reachability analysis, such as [21].

The fourth case is overtaking. If there are multiple obstacles detected on the ego lane, the vehicle should make the decision whether it should a) overtake the first obstacle, merge back to the ego lane, and the perform the same routine again for the second obstacle, or b) just overtake all obstacles at once. We ensure that the vehicle will not get in a deadlock situation in which it can merge into the lane, but not out of the lane to overtake the second obstacle. We first compute the distance between the furthermost part of the first obstacle and the nearest part of the second obstacle. If this distance allows the in and out maneuver, the vehicle computes a trajectory that overtakes the first obstacle, then returns to its lane, before repeating the same routine. If the gap is too small, then the vehicle overtakes both obstacles at once, and determines the next action between the second and third obstacles. In this work, we consider the minimum distance between obstacles to be a conservative estimate of $6 \times \frac{tan(\delta_{max})}{L}$.

The fifth case is returning to the nominal driving lane. Merging back to the ego lane is performed by imposing a large amount of terminal cost to the optimizer when the vehicle is close to the end of the obstacle. In this work, we only impose the terminal cost when the furthermost part of the obstacle in consideration is within the planning horizon, i.e. $v_k \times N \times \Delta t_k$.

VI. Results

We conduct simulations with Stage within the ROS framework [22]. We use a previously mapped area of One North in Singapore. The simulation set up mimics the single SICK LMS-151 that has been installed on our autonomous vehicle [20], with 180^o the field-of-view, $0,5^o$ its precision, and $50m$ the sensing range. The simulated vehicle has a maximum cruising speed of $5m/s$, a maximum approach speed $3m/s$, and a maximum overtaking speed of $2m/s$. Laser scan points in the ego lane are first clustered, a rectangle is then fitted, and the blind spot frontier point extracted from the individual clusters. The MPC optimization is solved at $10Hz$, with 50 steps horizon, and $0.1s$ time step, with the code generated by FORCES Pro [23], a commercial code generator for optimization solvers.

A video that showcases the planner's capabilities can be accessed at https://youtu.be/INynNEw10OU. Snapshots of the different scenarios are shown in Fig. (8-11). Every scenario starts with the vehicle following an obstacle free path. The window of the left displays the visualization of the robot's perception and planning intentions. Blue lines indicate road boundaries, yellow lines indicate the centre line of the path, green line indicates the MPC plan. The dynamic virtual bumper is displayed as the polygon. Red points indicate obstacles that lie in either the ego lane or the opposing lane, while white points outline the rectangular obstacle clusters. The window on the right is the simulator. The red box is the ego vehicle and blue and yellow boxes are the obstacles. The green area is the coverage of the simulated LIDAR signal.

In the first scenario (Fig. 8), the vehicle must overtake three vehicles parked in parallel, in this case the length of the obstacle is first unknown to the vehicle. As described in Section IV, the vehicle will first move towards the center of the lane and assess the presence of an obstacle in the opposite lane, before it cuts into the opposite lane. When it gets enough information of the size of the obstacle it executes the overtaking maneuver and safely merges back into its own lane.

In the second scenario (Fig. 9), two obstacles are further apart from each other, but not enough for the vehicle to merge in and out of the lane. Similar to previous scenario, the vehicle first moves towards the centre of the lane to get more visibility of the opposite lane, to asses the presence of obstacles in the opposite lane, and to get better information of the length of the unknown obstacle. As soon as it detects the rear end of the second obstacle, it computes the distance between the obstacles to determine the course of action, but still conservatively cuts to the other lane making sure that it can come back to its lane. As soon as the vehicle does not enough clearance to return to the original lane, and there are no obstacles detected in the opposite lane, it executes the overtaking maneuver and later merges back to the original lane safely.

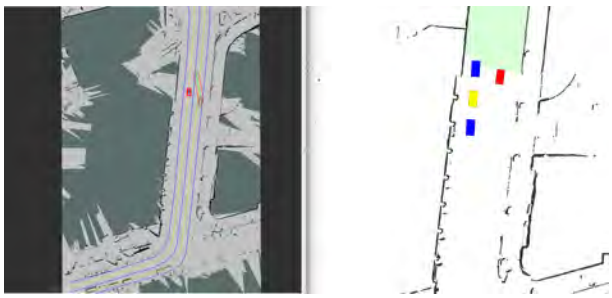

Fig. 8. Snapshot of simulation scenario 1.

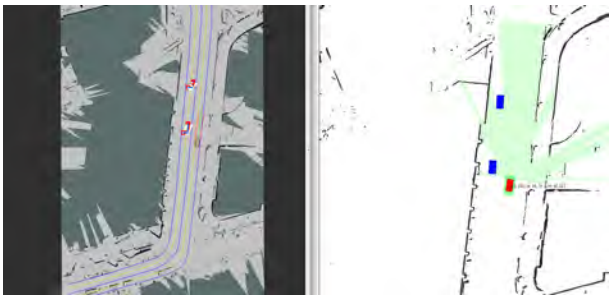

Fig. 9. Snapshot of simulation scenario 2.

The third scenario (Fig. 10) is similar to the second scenario. However, in this scenario, the obstacles are located further apart. As soon as two vehicles are detected separately, and the distance between the vehicles is verified to be enough for the vehicle to merge into the lane and come back out, the vehicle will find a plan to merge back into the lane. As soon as the first obstacle disappears from the field of view of the autonomous vehicle, the overtaking problem becomes similar to scenario one, but with the vehicle starting slightly out of the original lane. The vehicle is then able to perform the overtaking maneuver safely.

The fourth scenario (Fig. 11) is the same as the third scenario, but with an obstacle in the opposite lane. Similar to the third scenario, as soon as the autonomous vehicle detects that the two obstacles are far enough, i.e. that it has enough clearance to merge in an out of the lane, it executes the overtaking maneuver. When it detects the obstacle in the opposite lane, it responds by merging back into the lane, waiting behind the second obstacle. As soon as the opposite lane is clear, the vehicle can execute an overtaking maneuver similar to scenario one.

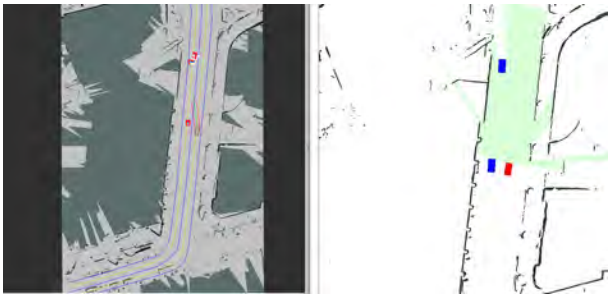

Fig. 10. Snapshot of simulation scenario 3.

Fig. 11. Snapshot of simulation scenario 4.

VII. Conclusion and Future Works

In this paper we have investigated the problem of overtaking unexpected obstacles on a two-way street in an urban environment. We have proposed a Receding Horizon formulation that takes into account the blind spot caused by occluding obstacles and maximizes visibility. We also have designed a higher-level state machine, which ensures that the autonomous vehicle can generate safe and optimal overtaking trajectories. We have demonstrated the capabilities of the method in a simulation environment.

In our simulation, we have taken a very conservative approach in designing our behaviour planner, i.e. the system will default to return to its ego lane when an obstacle is detected in the opposite lane and wait until the obstacle in the opposite lane has been cleared to proceed with the overtaking maneuver. This may pose a problem when an obstacle suddenly appear or disappear from the vehicle's sensing coverage. In this case, a significant change in the cost and constraints may result in significant change in the optimal inputs, or high slack term in the optimization routine. We are also aware that we have are yet to consider the possibility of the static obstacle in the ego lane starts moving while the system is executing the overtaking maneuver, this problem also extends to overtaking unusually slow vehicles in the ego lane.

In the future, we plan to perform real world experiments in the aforementioned one-north district in Singapore, by integrating the planner into our overall architecture described in [20]. We hope that by gaining real-world experimental data and considering the risks associated with breaking the traffic rules to overtake unexpected obstacles, we can further refine and increase the sophistication of the current algorithm.

Acknowledgment

This research was supported by the Future Urban Mobility project of the Singapore-MIT Alliance for Research and Technology (SMART) Center, with funding from Singapore's National Research Foundation (NRF).

References

[1] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416–442, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X15003447

[2] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, March 2016.

[3] S. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang, "Perception, Planning, Control, and Coordination for Autonomous Vehicles," *Machines*, vol. 5, no. 1, p. 6, 2017. [Online]. Available: http://www.mdpi.com/2075-1702/5/1/6

[4] M. Althoff, O. Stursberg, and M. Buss, "Model-Based Probabilistic Collision Detection in Autonomous Driving," vol. 10, no. 2, pp. 299–310, 2009.

[5] R. Hayashi, J. Isogai, P. Raksincharoensak, and M. Nagai, "Autonomous Collision Avoidance System by Combined Control of Steering and Braking using Geometrically Optimised Vehicular Trajectory," *Vehicle System Dynamics*, vol. 50, no. Suplement, pp. 151–168, 2012.

[6] C. Guo, K. Kidono, and M. Ogawa, "Learning-based trajectory generation for intelligent vehicles in urban environment," *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2016-Augus, no. Iv, pp. 1236–1241, 2016.

[7] T. Shamir, "How should an autonomous vehicle overtake a slower moving vehicle: Design and analysis of an optimal trajectory," *IEEE Transactions on Automatic Control*, vol. 49, no. 4, pp. 607–610, 2004.

[8] J. Baber, J. Kolodko, T. Noel, M. Parent, and L. Vlacic, "Cooperative autonomous driving: intelligent vehicles sharing city roads," *IEEE Robotics & Automation Magazine*, vol. 12, no. 1, pp. 44–49, 2005.

[9] A. G. Cunningham, E. Galceran, R. M. Eustice, and E. Olson, "Mpdm: Multipolicy decision-making in dynamic, uncertain environments for autonomous driving," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 1670–1677.

[10] J. Schlechtriemen, K. P. Wabersich, and K. D. Kuhnert, "Wiggling through complex traffic: Planning trajectories constrained by predictions," *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2016-Augus, no. Iv, pp. 1293–1300, 2016.

[11] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, Jun. 2011.

[12] F. Wang, M. Yang, and R. Yang, "Conflict-probability-estimation-based overtaking for intelligent vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 366–370, 2009.

[13] S. D. Cairano, U. V. Kalabi, and K. Berntorp, "Vehicle Tracking Control on Piecewise-Clothoidal Trajectories by MPC with Guaranteed Error Bounds," no. Cdc, 2016.

[14] G. Schildbach, M. Soppert, and F. Borrelli, "A Collision Avoidance System at Intersections using Robust Model Predictive Control," *IEEE Intelligent Vehicles Symposium (IV)*, no. Iv, pp. 1–6, 2016.

[15] M. Obayashi, K. Uto, and G. Takano, "Appropriate Overtaking Motion Generating Method using Predictive Control with Suitable Car Dynamics," no. Cdc, 2016.

[16] B. Yi, S. Gottschling, J. Ferdinand, N. Simm, F. Bonarens, and C. Stiller, "Real time integrated vehicle dynamics control and trajectory planning with MPC for critical maneuvers," *IEEE Intelligent Vehicles Symposium*, no. Iv, pp. 584–589, 2016.

[17] W. Schwarting, J. Alonso-Mora, L. Paull, S. Karaman, and D. Rus, "Parallel Autonomy in Automated Vehicles: Safe Motion Generation with Minimal Intervention," in *IEEE International Conference on Robotics and Automation*, 2017.

[18] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha - A local, continuous method," *2014 IEEE Intelligent Vehicles Symposium (IV)*, vol. 35, no. April, pp. 450–457, 2014.

[19] D. Lam, C. Manzie, and M. Good, "Model predictive contouring control," in *49th IEEE Conference on Decision and Control (CDC)*, Dec 2010, pp. 6137–6142.

[20] S. D. Pendleton, H. Andersen, X. Shen, Y. H. Eng, C. Zhang, H. X. Kong, W. K. Leong, M. H. Ang, and D. Rus, "Multi-class autonomous vehicles for mobility-on-demand service," in *2016 IEEE/SICE International Symposium on System Integration (SII)*, Dec 2016, pp. 204–211.

[21] S. D. Pendleton, W. Liu, H. Andersen, Y. H. Eng, E. Frazzoli, D. Rus, and M. H. Ang, "Numerical approach to reachability-guided sampling-based motion planning under differential constraints," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1232–1239, July 2017.

[22] M. Quigley, K. Conley, B. Gerkey, J. FAust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Mg, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009.

[23] embotech. (2017, April) FORCES Pro. [Online]. Available: https://www.embotech.com/FORCES-Pro