Following A Dynamic Object Through A Transient Response Adjustable MPC

Sanghyun Hong^{*†}, Justin Miller^{*}, and Jianbo Lu^{*}

Abstract—Robots can help humans fulfill various tasks in efficient and safe manners. For example, robots can navigate while loaded with packages to assist humans in delivery tasks, and they can also collaborate with Autonomous Vehicles (AV) to reach areas that AVs cannot drive. As robots are expected to interact with humans, vehicles, and other robots, following these dynamic objects is considered a crucial functionality. This paper will propose a mobile robot motion controller for following a dynamic object based on Model Predictive Control (MPC) that can reflect the dynamics of object for an appropriate response speed and ensure safety of mobile robot against predicted risks. The MPC is formulated in a way that is simple enough to compute control commands in real time without an optimization solver but can accomplish the above tasks successfully. The proposed motion controller is demonstrated through extensive simulations and real-time experiments of a test mobile robot.

I. INTRODUCTION

Human-robot interaction has gained increasing attention, allowing humans to fulfill tasks quickly and safely through assistance of robots. In order for mobile robots to interact with humans, a crucial functionality is following a person in a way that ensures efficient human-robot interaction. For example, if a mobile robot follows a delivery person with packages on it, following distance should be within a reasonable range for delivery efficiency and the mobile robot should be able to regulate its maneuvers to ensure safety of itself and its packages. In addition to human-robot interaction, following functionality is also important when a mobile robot needs to collaborate with another mobile robot, e.g., mobile robot platooning for mass package delivery.

To follow an object, mobile robots need an object tracking algorithm that recognizes the location of the object and a motion control algorithm that creates maneuvers that will follow the object in an appropriate manner. The motion control algorithm should be able to follow an object with a reasonable distance while ensuring safety of maneuver and avoiding collisions with obstacles. There have been various motion control algorithms proposed to follow people. Lee et al. [1] used Dynamic Window Approach (DWA) within the Robot Operating System (ROS) navigation stack. A drawback to that approach is that the prediction of the trajectory is limited because of a short prediction time horizon. Moreover, safe maneuvering of the mobile robot and correct following of the planned path were not taken into consideration in its baseline algorithm. They had to implement additional control

schemes to compensate for limitations of DWA. Doisy et al. [2] utilized a path following algorithm that applied a discontinuous feedback control law proposed by Morin et al. [3], which is known as Sliding Mode Control (SMC). Shanee et al. [4] implemented Proportional Control (P-Control) along with the SMC for path following algorithm in order to follow a person with a pre-defined angle. However, their approach considered neither the response speed to the motion of persons nor safe maneuvering of mobile robots. Chen et al. [5] developed a vision-based person following in which a simple proportional control was used without accounting for safe maneuvering of mobile robots and collisions with obstacles. Olmedo et al. [6] developed a people following mobile robot with laser scanners, but it is based on PID control which does not take safety of the mobile robot and optimality of control into consideration. Although Leigh et al. [7] extended the methodology utilizing laser scanners so that mobile robots can track legs of person, they employed the same PID controller for following persons.

In following a highly dynamic target, e.g., a human, it is important to find optimal control commands that balance the trade-off between navigation cost (e.g., time and actuation) and safety (e.g., collision avoidance and safe cornering) before any potential risk is imminent. Therefore, in this paper, we will apply Model Predictive Control (MPC) to develop a motion control algorithm for following a dynamic object. There are several approaches used for motion control of mobile robots, e.g., Dynamic Window Approach (DWA) [8] and Trajectory Rollout (TR) [9], that are based on the predictive optimal control framework. However, they mainly focused on reaching a goal without taking into account safe maneuvering of mobile robots, response speed, or tight path following.

This paper proposes a motion controller for following a dynamic object based on MPC that is adaptable to reflect the dynamics of the object while ensuring the safety of mobile robots. The MPC is formulated in a way that is simple enough to compute control commands in real time without an optimization solver while also achieving the above objectives successfully. The proposed motion controller is demonstrated through extensive simulation and experimental tests.

II. PROBLEM DEFINITION

The problem to be addressed in this paper is related to how a mobile robot is following a dynamic object. The solution to this problem has application in various domains, e.g., package delivery and personal assistance. Consider the

[†] Corresponding author

^{*} S. Hong, J. Miller, and J. Lu are with Research & Advanced Engineering, Ford Motor Company, Dearborn, MI, USA (e-mail: {shong7; jmill597; jlu10}@ford.com).

case shown in Fig. 1, where a blue mobile robot is tasked to follow the orange mobile robot by tracking its relative location. Once the blue robot detects the orange robot, it sets a goal with an offset d_{off} from the orange robot, as indicated by the yellow circle in Fig. 1. Then, the blue mobile robot follows the goals that are published whenever a time period or minimum displacement is met as the orange robot moves.

The blue robot computes control commands, u , in the form of linear and angular velocities that will depend on the distance, $d(k)$, from the blue robot to the goal at time k . In Fig. 1, the length of arrows in Fig. 1 represents the magnitude of the computed linear velocity. Additionally, there are constraints on the linear and angular accelerations, i.e. the upper bound on the magnitude of acceleration, to prevent excessively aggressive motion that can cause a dangerous situation. In order to follow a highly dynamic object, e.g., a

Fig. 1: A Mobile Robot Following A Dynamic Object

human, that changes maneuvers continuously and drastically, the wheeled mobile robot should be able to account for the response speed for the highly dynamic motion of the object. In addition, for the sake of safety, the mobile robot should have the capability to avoid collisions with obstacles and loss of control.

III. FOLLOWING MOTION CONTROL WITH MODEL PREDICTIVE CONTROL

When following a dynamic object, if the object changes maneuvers, a mobile robot needs to respond to the changes of motion by accelerating linear and angular velocities so that it does not lose track of the dynamic object. In addition, if a dynamic object stays at a steady state, a mobile robot also needs to find appropriate navigation velocities to remain within an appropriate distance from the object. Meanwhile, if any potential risks are anticipated, the mobile robot should prioritize safety and compute control commands to prevent dangerous situations at the cost of causing the mobile robot to fall behind. This implies that there exists trade-offs between following distance and safety, thus leading to the need for an optimal control method. In addition, taking action preemptively against potential risks is the best way to ensure safety, which requires looking ahead. This leads us to prediction of motion in the motion control scheme.

Therefore, we will use Model Predictive Control (MPC), also known as receding horizon optimal control, to develop a motion control algorithm that follows a dynamic object with safety ensured. With the MPC framework, a constrained optimal control problem is solved at time step k to obtain a sequence of optimal control commands, i.e. $u(k), \dots, u(k+N)$, where N represents the prediction time horizon. Once the optimal control commands are computed, only the first command $u(k)$ will be applied to mobile robot to generate a desired maneuver. Then, the above procedures are repeated at every time step afterwards.

In this paper, we formulate the constrained optimal control problem for the MPC-based motion controller by considering safe motion of mobile robot, tight path following, collision avoidance, and realistic accelerations, as follows.

$$\min_U h(\{s(j)\}_{j=k_c}^{k_c+N}, U) \quad (1a)$$

$$\text{subj. to } s(k+1) = f(s(k), u(k)) \quad (1b)$$

$$\ddot{\psi}^{lb} \leq \ddot{\psi}(k_c + j) \leq \ddot{\psi}^{ub} \quad (1c)$$

$$a_x^{lb} \leq a_x(k_c + j) \leq a_x^{ub} \quad (1d)$$

$$a_y^{lb} \leq a_y(k_c + j) \leq a_y^{ub} \quad (1e)$$

$$\text{for } j = 0, 1, \dots, N,$$

where k_c represents the time step at which the above problem is solved and N represents the time horizon for prediction. In (1), the constrained optimal control problem minimizes a cost function $h(\cdot)$ in (1a) which is a function of a predicted sequence of system states $\{s(j)\}_{j=k_c}^{k_c+N}$ and control inputs $U = \{u(j)\}_{j=k_c}^{k_c+N} = [u(k_c), \dots, u(k_c + N)]$. The system, i.e., a mobile robot, state $s(k)$ evolves based on the system dynamic equation $f(\cdot)$ in (1b). Accelerations in linear and rotational motion, i.e., $a_x(k)$, $a_y(k)$ and $\ddot{\psi}(k)$, are constrained by lower and upper bounds in (1c)-(1e), where the superscripts lb and ub represent lower bound and upper bound, respectively. As shown in Fig. 2, the subscripts x and y represent the longitudinal and lateral directions in the local frame of mobile robot, and ψ represents the rotational motion about z axis pointing toward readers, i.e. yaw motion. The

Fig. 2: A Global Frame and A Local Frame of Mobile Robot

global frame $OXYZ$ in Fig. 2 is an inertial frame that is fixed with a constant orientation.

A. Mobile Robot System Dynamics

In this subsection, we define the state s for the mobile robot in (1b) that consists of the coordinate X and Y in the global frame, yaw/heading angle ψ , longitudinal velocity v_x , lateral velocity v_y , and yaw rate $\dot{\psi}$, i.e., $s =$

$[X, Y, \psi, v_x, v_y, \dot{\psi}]^\top$. With the sampling time τ , the dynamic state space representation is constructed as follows.

$$\begin{bmatrix} X(k+1) \\ Y(k+1) \\ \psi(k+1) \\ v_x(k+1) \\ v_y(k+1) \\ \dot{\psi}(k+1) \end{bmatrix} = \begin{bmatrix} X(k) + \tau v_X(k) \\ Y(k) + \tau v_Y(k) \\ \psi(k) + \tau \dot{\psi}(k) \\ v_x(k) + \tau \dot{v}_x(k) \\ v_y(k) + \tau \dot{v}_y(k) \\ \dot{\psi}(k) + \tau \ddot{\psi}(k) \end{bmatrix}, \quad (2a)$$

where

$$\begin{bmatrix} v_X(k) \\ v_Y(k) \\ \dot{v}_x(k) \\ \dot{v}_y(k) \end{bmatrix} = \begin{bmatrix} v_x(k) \cos \psi(k) - v_y(k) \sin \psi(k) \\ v_x(k) \sin \psi(k) + v_y(k) \cos \psi(k) \\ a_x(k) + v_y(k) \dot{\psi}(k) \\ a_y(k) - v_x(k) \dot{\psi}(k) \end{bmatrix}. \quad (2b)$$

In the above equations, v_X and v_Y represent the velocities in the X and Y direction of global frame, respectively. In addition, $\dot{v}_x$ and $\dot{v}_y$ represent the time derivatives of the longitudinal velocity v_x and lateral velocity v_y of the mobile robot, respectively. Readers are referred to [10] for derivation of the above dynamic equations.

Note that this paper will use a nonholonomic mobile robot in *Simulation Results* and *Experimental Results*, and therefore v_y and $\dot{v}_y$ in (2) will be set to zeros.

B. Cost Function

The cost function to be minimized is defined with the predicted states and control commands as follows.

$$h(\{s(j)\}_{j=k_c}^{k_c+N}, U) = \omega_o \max(\{h_o(d_o(i))^2\}_{i=k_c}^{k_c+N}) \quad (3a)$$

$$+ \omega_{ay} \max(\{a_y(i)^2\}_{i=k_c}^{k_c+N}) \quad (3b)$$

$$+ \omega_g d_g(k_c + N)^2 \quad (3c)$$

$$+ \omega_p \sum_{j=0}^N d_p(k_c + j)^2, \quad (3d)$$

where ω_o , ω_{ay} , ω_g , and ω_p are weights.

The first cost term in (3a) enables avoiding collisions with obstacles. The function h_o is a function that is inversely proportional to $d_o(k)$ which is the distance at time k from the mobile robot to an obstacle, as shown in Fig. 3a. Therefore, the term in (3a) accounts for the location where the mobile robot navigates closest to an obstacle along the predicted trajectory. The predicted trajectory is obtained from the sequence of the first and second state variables, X and Y . In Fig. 3a, the magenta line represents the path planned from the mobile robot to a goal, and the green line is the predicted trajectory.

In order to ensure the safety of the mobile robot, we consider the largest lateral acceleration $a_y(\cdot)$ calculated along the predicted trajectory. If the mobile robot does not slow down in a sharp turn while following an object, it may lose

contact with the ground and slide. Furthermore, if the mobile robot follows an object with packages loaded, it may cause the packages to fall when taking a sharp turn that generates a large lateral acceleration. Therefore, the lateral acceleration is included in the cost function for safe maneuvering of the mobile robot. With a nonholonomic mobile robot, we can obtain a_y from the fourth element in (2b) by substituting zero to $\dot{v}_y$ as $a_y = v_x \dot{\psi}$.

To head for a goal, the distance to the goal d_g is included in the cost, as presented in (3c). Note that the distance to the goal at the end of the predicted trajectory is used, as shown in Fig. 3b, and the distance is not a straight airline distance to account for the distance required to navigate without collision with obstacles.

The fourth term in (3d) contains the distance from the mobile robot to the planned path d_p , as shown in Fig. 3c, and therefore, this cost term is utilized for tight path following. The path is planned by a path planning algorithm, e.g., A^* algorithm [11], and the details are not discussed herein since it is beyond the scope of this paper. If a path is planned periodically to reflect a dynamic environment, the planned path can be used as guidance for navigation of mobile robot. Therefore, following a planned path tightly can be helpful for the mobile robot to go around obstacles.

C. Accelerations to Track Constant Reference Velocities

In general, the decision variables in (1) would be the sequence of accelerations in linear and rotational motion, $U = [\{a_x(j)\}_{j=k_c}^{k_c+N}, \{a_y(j)\}_{j=k_c}^{k_c+N}, \{\ddot{\psi}(j)\}_{j=k_c}^{k_c+N}]$, which is a vector in the dimension of $3(N+1)$. But, in this paper, we utilize the steady state reference tracking control [12] to define the accelerations such that the mobile robot can converge to the constant reference velocities, v_x^{ref} , v_y^{ref} , and $\dot{\psi}^{ref}$, over the prediction time horizon $\tau_h = N\tau$, as follows.

$$\begin{bmatrix} a_x(k) \\ a_y(k) \\ \dot{\psi}(k) \end{bmatrix} = \begin{bmatrix} K_x(v_x^{ref} - v_x(k)) \\ K_y(v_y^{ref} - v_y(k)) \\ K_\psi(\dot{\psi}^{ref} - \dot{\psi}(k)) \end{bmatrix}, \quad (4)$$

where the feedback gains are

$$K_i = \frac{1 - p_i}{\tau}, \text{ for } i \in \{x, y, \psi\}. \quad (5)$$

With the above approach, the accelerations are constrained to functions of the constant reference velocities. In the definition of gain in (5), p_i represents the closed loop pole that determines the rate of convergence, and its absolute value should be $|p_i| < 1$ for stability. As the value of p_i becomes smaller, the velocity of the mobile robot converges to the reference one faster.

By substituting (4) and (5) into (1), the decision vector becomes $U = [v_x^{ref}, v_y^{ref}, \dot{\psi}^{ref}]$, which is in the 3-dimensional space, since $v_x(k)$, $v_y(k)$, and $\dot{\psi}(k)$ in (4) are obtained as the state variables. Therefore, the constrained optimal control problem in (1) can be solved with a significantly reduced computational cost.

If p_i is zero in (5), the mobile robot achieves the constant reference velocity immediately after one sampling time τ ,

Fig. 3: Cost Function

and the reference velocity is maintained for the entire prediction time horizon τ_h since the accelerations become zero after one sampling time. Although the immediate convergence to the reference velocities with $p_i = 0$ is not realistic, it has been assumed previously in the existing motion control algorithms, e.g., Dynamic Window Approach (DWA) [8] and Trajectory Rollout (TR) [9]. However, this paper will use nonzero p_i in simulations and experiments.

With the accelerations constrained to converge to constant reference velocities, it is found out that the prediction time horizon can be used to adjust transient response in following a dynamic object, as presented below. Therefore, the prediction time horizon provides a way to determine how quickly a mobile robot responds to motion of a dynamic object.

Theorem 3.1: Suppose an MPC defined in the form of (1) whose cost function includes the distance to goal at the end of predicted trajectory, $d_g(k_c + N)$. If the linear accelerations, a_x and a_y , are constrained to track constant reference velocities, v_x^{ref} and v_y^{ref} , the time horizon for prediction, $\tau_h = \tau N$, acts as the time constant to determine transient response.

Proof: The above statement can be proved by direct proof. Without loss of generality, consider a mobile robot heading for a static goal on a straight line in the X direction of the global frame, and it can generate the acceleration $a_x(k)$ to converge the constant reference velocity v_x^{ref} immediately. In addition, cost terms other than the distance to a goal and acceleration constraints are excluded. Then, the MPC is formulated as follows.

$$\begin{aligned} \min_{v_x^{ref}} \quad & d_g(k_c + N) \\ \text{subj. to} \quad & \begin{bmatrix} X(k+1) \\ v_x(k+1) \end{bmatrix} = \begin{bmatrix} X(k) + \tau v_x(k) \\ v_x(k) + \tau a_x(k) \end{bmatrix}, \\ & \text{where } a_x(k) = \frac{1}{\tau}(v_x^{ref} - v_x(k)). \end{aligned}$$

If the above problem is solved at every time step with sampling time τ , we always obtain the optimal constant reference velocity v_x^{ref} such that $v_x^{ref} \tau_h = d_g(k)$, where

$\tau_h = \tau N$ and $d_g(k)$ is the distance to goal at the time k when the above problem is solved.

In the continuous time t domain, we can rewrite as follows.

$$v_x^{ref} \tau_h = d_g(t),$$

and since $v_x^{ref} = \dot{X}(t)$,

$$\dot{X}(t) = \frac{d_g(t)}{\tau_h} = \frac{X_g - X(t)}{\tau_h}, \quad (7)$$

where $X(t)$ and X_g represent the coordinates of mobile robot and goal in the global frame, respectively.

Applying the Laplace Transform to (7) with the assumption of zero initial condition yields,

$$\begin{aligned} s\mathcal{X}(s) &= \frac{1}{\tau_h} \frac{X_g}{s} - \frac{1}{\tau_h} \mathcal{X}(s) \\ \Rightarrow (s + \frac{1}{\tau_h})\mathcal{X}(s) &= \frac{1}{\tau_h} \frac{X_g}{s} \\ \Rightarrow \mathcal{X}(s) &= \frac{X_g/\tau_h}{s(s + 1/\tau_h)} \end{aligned} \quad (8)$$

where the Laplace Transform of $X(t)$ is represented by $\mathcal{X}(s) = \mathcal{L}\{X(t)\}$ and s represents the complex variable.

Then, the Inverse Laplace Transform of (8) yields

$$X(t) = X_g(1 - e^{-t/\tau_h}). \quad (9)$$

By the definition of time constant, the time at which $X(t)$ reaches around $(1 - 1/e) \times 100 \approx 63.2\%$ of X_g is the prediction time horizon τ_h , as shown in (9). ■

D. Solving the MPC

In order to solve the MPC in (1), the method used in DWA and TR are applied in this paper. Therefore, n_i values are sampled from each decision variable, and a search space is constructed with all possible combinations of the sampled values. Then, by searching for the grid point in the search space that generates the smallest cost in (3), the optimal combination of decision variable values can be found.

In this paper, due to (4), the decision variables are $U = [v_x^{ref}, v_y^{ref}, \psi^{ref}]$. Therefore, for example, we sample n_{v_x} values spacing the range of $[v_x(k) - \tau_h a_x^{lb}, v_x(k) + \tau_h a_x^{ub}]$

for the decision variable v_x^{ref} . In other words, the i^{th} sample for v_x^{ref} is

$$v_x - \tau_h a_x^{lb} + \frac{\tau_h (a_x^{lb} + a_x^{ub})}{n_{vx}} i.$$

Likewise, n_{vy} and n_{ψ} values are sampled for the decision variables v_y^{ref} and $\dot{\psi}^{ref}$, respectively. If a nonholonomic mobile robot is used, we sample values from v_x^{ref} and $\dot{\psi}^{ref}$ only, and the optimal decision variables v_x^* and $\dot{\psi}^*$ are found by evaluating costs for the sampled values.

With the above approach, the MPC in (1) can be solved without an optimization solver. Most optimization solvers assume the cost function is differentiable, and therefore if we do not use an optimization solver, a non-differentiable cost function can be formulated. In addition, we can guarantee a global minima within the search space. If we define an MPC with many decision variables and many sample values are required for the decision variables, this approach is not feasible in real time and an optimization solver may be necessary. But, we have the 3 dimensional decision vector, and, in practice, with mobile robots, we do not have to search for v_x^{ref} values at very fine resolutions, e.g., 0.01 m/s. Therefore, we can solve the MPC in (1) in real time with a reasonably small number of samples for the decision variables.

IV. SIMULATION RESULTS

In this section, the proposed motion controller will be evaluated in simulation using a Robot Operating System (ROS) environment. The dynamic object following software stack is implemented as an extension of the ROS navigation stack [13] with the motion controller algorithm being replaced by the one proposed in this paper. As a result of using this framework, the *Path Planner* [14] in Fig. 4 plans a path from a mobile robot to a goal on 2-dimensional costmaps created by the *costmap_2d* ROS package [15] with a fast interpolated navigation function.

In the simulation, we use an ideal nonholonomic mobile robot platform on a 2-dimensional surface to verify performance of the proposed motion controller without extrinsic influences, e.g., roll motion and time lagging of actuator. Therefore, the 2-dimensional mobile robot platform is im-

Fig. 4: Motion Control for Dynamic Object Following

plemented as a ROS node, as shown in Fig. 4, based on the dynamic vehicle model in (2) with $v_y = 0$ and $\dot{v}_y = 0$. The mobile robot in simulation takes in the optimal commands v_x^* and $\dot{\psi}^*$ computed by the proposed motion controller and yields the pose (X, Y, ψ) in the global frame along with

its velocity states v_x and $\dot{\psi}$. The states of mobile robot are updated every 0.05 seconds. The velocity states v_x and $\dot{\psi}$ are provided to the motion controller as feedback signals. A dynamic object is simulated through the *Goal Publisher* in Fig. 4 that publishes goals moving with specified velocities every 0.1 seconds. Note that a localization algorithm does not run in the proposed following software stack since a prior map is not necessary for following a dynamic object in this paper. The parameters for the motion controller are listed in Table I, and the motion control algorithm computes the control commands at an update frequency 5 Hz.

TABLE I: Motion Controller Parameters

Parameter	Value	Parameter	Value
p_x	0.2	ω_o	0.0001
p_ψ	0.2	ω_{ay}	1.0
τ	0.1 [s]	ω_g	0.001
$\ddot{\psi}^{ub}$	4.5 [rad/s]	ω_p	0.01
$\ddot{\psi}^{lb}$	-4.5 [rad/s]	n_ψ	60
a_x^{ub}	0.5 [m/s ²]	n_{vx}	30
a_x^{lb}	-0.5 [m/s ²]		

A. Prediction Time Horizon as Time Constant

As discussed previously, with the proposed motion control formulation, the prediction time horizon τ_h acts as the time constant, and therefore we can tune τ_h to adjust the rate of convergence to a goal, as shown in Fig. 5. In this simulation,

Fig. 5: Different rate of convergence to a goal due to different prediction time horizon

the goal was static and located at 3 m in front of the mobile robot for the four different τ_h values, 1.5, 2.0, 2.5, and 3.0 seconds.

As shown in Fig. 5, with the smaller τ_h , the mobile robot accelerates v_x harder to reach the goal faster. The time steps when the mobile robot navigates 63.2% of the distance to goal are around 0.9, 1.2, 1.7, and 2.0 seconds, respectively. The values of τ_h do not match the measured time steps exactly, but this simulation demonstrates that τ_h can be used to adjust the rate of convergence to a goal.

In the simulation presented in Fig. 6, the mobile robot follows a moving object with two different prediction time

horizons τ_h . The positions of the moving object are published as goals in the ROS environment. The speed of the object is 0.0 m/s at first, and it reaches 1.0 m/s immediately after around 2.0 seconds has passed. This abrupt speed change simulates following a highly dynamic object, e.g., a human.

Fig. 6: Following a highly dynamic object with different prediction time horizon

In Fig. 6, the mobile robot with $\tau_h = 2.5$ falls behind the one with $\tau_h = 1.5$ at the end of simulation. This is because the velocity v_x of the mobile robot increases faster to catch up with the dynamic goal more quickly with $\tau_h = 1.5$ than $\tau_h = 2.5$, as shown in Fig. 7.

Fig. 7: Longitudinal velocity and yaw rate generated with different prediction time horizon

B. Tight Path Following

The proposed motion controller takes tight path following into consideration through the cost term in (3d). The tightness of the path following can be adjusted by tuning the weight ω_p . Fig. 8 presents the trajectories of the mobile robot and goal of a simulation in which the mobile robot moves straight first and starts taking left turn to follow the dynamic goal with $v_x = 2.0$ m/s and $\dot{\psi} = 0.7$ rad/s. In order to verify the performance of tight path following, the safe maneuver cost term in (3b) is set to zero. In Fig. 8, the orange line represents the last path in the simulation with $\tau_h = 2.5$

Fig. 8: Following a dynamic object with tight path following considered

that is planned from the mobile robot to the moving goal. The blue circles represent the positions of the mobile robot without the path following term in (3d), and therefore, the mobile robot mostly moves over the points that the dynamic goal has passed. But, with the tight path following cost term, the mobile robot, cyan circles in Fig. 8, takes a smaller turn than without considering tight path following since it tries to follow the planned path which is almost a straight line to the goal.

C. Safe Maneuver

The lateral acceleration a_y is included as a cost term in (3b) to generate safe maneuvers for the mobile robot while taking turns. In the simulation with $\tau_h = 2.5$ in Fig. 9, the mobile robot undergoes the same scenario as in Fig. 8. The tight path following cost term is used, and therefore the mobile robot tries to create a sharper turn than the dynamic object.

Fig. 9: Following a dynamic object with mobile robot's safe maneuver considered

In Fig. 9, the trajectory of the mobile robot with the a_y cost term in (3b) is similar to the one without it. But, with the safe maneuver considered, the mobile robot slows down to decrease the velocity v_x after about 5 seconds although it generates a similar yaw rate $\dot{\psi}$, as shown in Fig. 10.

Fig. 10: Longitudinal velocity and yaw rate generated when considering mobile robot's safe maneuver

V. EXPERIMENTAL RESULTS

In the simulation, the proposed motion controller is implemented in a ROS framework and the mobile robot is modeled as a separate ROS node. Therefore, experimental tests can be conducted by simply replacing the *Mobile Robot* in Fig. 4 with a real mobile robot platform.

A. Mobile Robot Platform

The test robot is built on the Clearpath Jackal differential-drive wheeled platform shown in Fig. 11. The base platform receives linear and angular velocity commands and provides odometry data from wheel encoders. The platform is outfitted with additional Lidar sensors. An RPLidar A3 2D Lidar is mounted on the front bumper and is used for obstacle detection. A Velodyne VLP-32C 3D Lidar is mounted on top and is used for target tracking.

Fig. 11: Test robot consisting of a Clearpath Jackal base with an RPLidar A3 on the front bumper and a Velodyne VLP-32C 3D Lidar on top.

B. Human Tracking and Goal Broadcasting

In the experimental tests, a human tracking algorithm replaces the *Goal Publisher* in Fig. 4 to broadcast goals for

the proposed motion controller. Human tracking is performed using the Leg Tracker ROS Package that is based on the work in [7]. A 2D laserscan is generated from the Velodyne and passed to the leg tracker ROS node. The leg tracker maintains unique labels for each human around the robot. The robot can be assigned a specific label ID to follow as its target. A direction vector is computed between the 2D pose of the target and the 2D pose of the robot within the global frame, i.e., the odometry frame in the ROS navigation stack. A goal is assigned as the location along that vector that is offset 1 m behind the target. Goals are repeatedly generated and provided to the motion controller at a rate of 7.5 Hz

C. Following Performance

Tests with the real mobile robot platform also utilize the same control update frequency 5 Hz and parameters listed in Table I as in the *Simulation Results* except for $\tau = 0.2$ second, and they demonstrate the finding that the prediction time horizon acts as the time constant. In Fig. 12, the percentage of travel distance is presented when the mobile robot is commanded to navigate to a static goal in front of it. The time steps when the mobile robot navigates 63.2% of the distance are around 1.7, 2.5, and 3.2 seconds for $\tau_h = 2.5$, $\tau_h = 3.5$, and $\tau_h = 4.5$, respectively. This implies that the prediction time horizon has a linear relationship with the time constant.

Fig. 12: Different rate of convergence to a goal due to different prediction time horizon

To validate the collision avoidance and following performance, the mobile robot was assigned to follow a person who is going over an obstacle, performing a zigzag maneuver, and turning around. In this test, the motion controller computed the following control commands with the prediction time horizon $\tau_h = 2.5$. As shown in the red box in Fig. 13a, the person walked in a straight-line path stepping over an obstacle while the mobile robot deviates from the person's trajectory to go around that obstacle. The gray dots in Fig. 13a and Fig. 13b represent occupied grid points in the costmap used in motion control. After avoiding collision with the obstacle, the mobile robot follows the person walking in a zigzag motion. In Fig. 13b, the mobile robot follows the person turning around in a tight space.

(a) Avoiding collision with an obstacle and following a person performing a zigzag maneuver

(b) Following a person turning around in a tight space

Fig. 13: Obstacle Avoidance and Following A Person

In addition, since the mobile robot takes safe maneuvers into consideration, it slows down if a sharp turn is required to follow the person, as shown in Fig. 14. The longitudinal velocity v_x decreases when the magnitude of yaw rate $\dot{\psi}$ increases significantly in the red boxes.

Fig. 14: Longitudinal velocity and yaw rate generated when a sharp turn is required to follow a person

VI. CONCLUSION

This paper proposes a motion controller to follow a dynamic object based on MPC. It is proved that the prediction time horizon acts as a time constant to determine the transient response in the proposed MPC formulation. Therefore, the proposed motion control algorithm can adapt to the dynamics of object by tuning the prediction time horizon. It also takes safe maneuvering and tight path following into consideration. The motion controller is formulated in a way that is simple enough to compute control commands in real time without an optimization solver but can still achieve the objectives successfully. The proposed motion controller are demonstrated through extensive simulation and experiments of a test mobile robot.

REFERENCES

- [1] B. Lee, J. Choi, C. Baek, and B. Zhang, "Robust human following by deep bayesian trajectory prediction for home service robots," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 7189–7195.
- [2] G. Doisy, A. Jevtic, E. Lucet, and Y. Edan, "Adaptive person-following algorithm based on depth images and mapping," in *In Workshop on Robot Motion Planning: Online, Reactive, and in Real-time, Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2012.
- [3] P. Morin and C. Samson, "Motion control of wheeled mobile robots," in *Springer Handbook of Robotics*, January 2008, pp. 799–826.
- [4] H. S. Shanee, K. Dror, O. Tal, and E. Yael, "The influence of following angle on performance metrics of a human-following robot," in *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, August 2016, pp. 593–598.
- [5] Zhichao Chen and S. T. Birchfield, "Person following with a mobile robot using binocular feature-based tracking," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2007, pp. 815–820.
- [6] N. A. Olmedo, H. Zhang, and M. Lipsett, "Mobile robot system architecture for people tracking and following applications," in *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, Dec 2014, pp. 825–830.
- [7] A. Leigh, J. Pineau, N. Olmedo, and H. Zhang, "Person tracking and following with 2d laser scanners," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 726–733.
- [8] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics Automation Magazine*, vol. 4, no. 1, pp. 23–33, March 1997.
- [9] B. P. Gerkey and K. Konolige, "Planning and control in unstructured terrain," in *In Workshop on Path Planning on Costmaps, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.
- [10] R. Rajamani, *Vehicle Dynamics And Control*. Springer Verlag, 2006.
- [11] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, July 1968.
- [12] P. J. Antsaklis and A. N. Michel, *Linear systems*. McGraw-Hill, 1997.
- [13] ROS.org. Navigation stack. [Online]. Available: <http://wiki.ros.org/navigation>
- [14] —. Navigation stack - navfn. [Online]. Available: <http://wiki.ros.org/navfn>
- [15] —. Navigation stack - costmap*2d. [Online]. Available: http://wiki.ros.org/costmap*2d