Adaptive Model Predictive Control of Wheeled Mobile Robots

Nikhil Potu Surya Prakash
3035299827

Tamara Perreault
3036327969

Trevor Voth
3033359760

Zejun Zhong
3036329035

***Abstract*—This paper presents an adaptive Model Predictive Control (MPC) strategy for a simple two-wheeled robot carrying unknown loads, with many creative applications such as distribution tasks. The method uses nonlinear system dynamics and nonholonomic constraints to generate a constrained finite time optimization problem (CTFOC) which uses the Python program "ipopt" to generate an optimal series of states and inputs. This is fed into an adaptive MPC program which reevaluates the CTFOC at each time step while also adapting to changing system parameters. The results are simulated with a basic 2D animation in Python.**

I. Introduction

Many research teams have considered methods for the control of simple two- wheeled robots. The simple unicycle kinematic model of this system provides an ideal starting point for many applications. Specifically, MPC algorithms have been used to consider nonholonomic constraints[1], to effectively control robots in actively changing environments[2], to generate effective tracking strategies[3], to control a robot that needs to push a large load[4], and many more applications. This paper builds on this previous work and presents a controller which considers nonholonomic constraints, tracks a point, and evaluates changing parameters, with the goal of controlling the robot to successfully carry a load of unknown size. The robot model has a rectangular base for wheel mounting and an otherwise cylindrical shape. There are various practical applications for such a robot, including distribution of items in warehouses or restaurants, cleaning, or sport [5]. For example, such a robot could pick up a ball of unknown size and carry it along an optimal trajectory to a goal. In a warehouse, such a robot could support the transport of items too awkward or heavy to be carried otherwise. These applications are particularly relevant under the current circumstances of the COVID-19 pandemic, where the number of people in a warehouse or restaurant is necessarily limited by health and safety standards, and sports are not being played as frequently in person.

Paper structure: In Section II, we discuss the nonlinear dynamics used to model the wheeled robot. In Section III, we present the MPC Algorithm used to control the robot along a trajectory. In Section IV, we explain the adaptive component of the MPC control. In Section V, we discuss the performance of the controller as modeled by simulations. In the Appendix, we show the results of the MPC algorithm with the known system parameters along with the adaptive MPC algorithm with unknown system parameters.

II. Dynamics of a wheeled mobile robot

In this section, the dynamic evolution of the states of the wheeled mobile robot will be presented. The mechanical structure of the wheeled mobile robot is such that it is driven as a differential drive with two wheels actuated by motor torques at the rear and a smooth caster in the front keeps the robot stable. The dynamics are based on the dynamic model of a unicycle with a nonholonomic constraint that the admissible velocity of the robot is only along its orientation, thereby implying that the velocity in the direction perpendicular to its orientation is zero. The mathematical structure of the robot is isomorphic to that of a knife edge actuated by a thrust for motion control along its heading and a torque along the vector perpendicular to the motion plane of the robot to control its orientation.Using the isomorphic structure between the dynamics of the wheeled mobile robot and the knife edge, the motor torques on the wheels of the robot can be easily calculated using a one-to-one transformation. Hence to simplify the analysis and notation, a dynamic knife edge model will be used in the rest of the paper. The dynamics are described using the Lagrangian formulation
The system in consideration is similar to a knife in structure but with more assumptions on the constraints that the frictional capacity available to keep the skate from moving in the direction perpendicular to its orientation.Further we assume that the frictional force along the direction of motion allow only pure rolling of the wheels and no slippage is allowed.
We define two frames, an inertial frame (with subscript A) and a body fixed frame (with subscript B) in our analysis.The transformation between these two frames is achieved through the yaw angle.The kinetic energy of the system with states $[x, y, \psi, \dot{x}, \dot{y}, \dot{\psi}]$ is given by

$$T = \frac{1}{2}m\dot{x}^2 + \frac{1}{2}m\dot{y}^2 + \frac{1}{2}J\dot{\psi}^2 \tag{1}$$

where x and y are the coordinates of the robot, ψ is its heading angle and the rest are their derivatives.
We assume that the motion of the knife edge is restricted to a plane with the same gravitational potential throughout and hence there will not be any change in the potential energy of the system as the knife edge moves.But this analysis can easily be extended to a case where gravitational potential need not be constant along the path. Since the other potentials remain constant, the kinetic energy will be equal to the Lagrangian $(L := T - V)$ of the system.

The constraint on the motion is given by the condition that the velocity in the direction j_B *i.e.*, along the direction perpendicular to its orientation is 0

$$-\dot{x}sin\psi + \dot{y}cos\psi = 0 \tag{2}$$

We control the system by giving two inputs on the system. A thrust R in the direction of the velocity and a moment M to change the orientation of the knife edge. The equations of motion are derived using the Lagrangian formulation by introducing Lagrangian multipliers (λ) since the system is nonholonomic. One can even use the Appelian approach using the acceleration energy to arrive at the equations of motion.The generalized forces are obtained by writing the virtual work done by the forces on the system. The generalized forces on the system can be derived by writing the virtual work (δW) done by the control inputs on the system.

$$\begin{aligned}\delta W =&(\delta x cos\psi + \delta y sin\psi)(R - bv) \\ &+ \delta\psi(M - c\dot{\psi})\end{aligned} \tag{3}$$

Where $v = \dot{x}cos\psi + \dot{y}sin\psi$ is the longitudinal velocity, b and c are the drag coefficients corresponding to the linear and angular motions respectively. Therefore the generalized forces Q_x , Q_y and Q_ψ are equal to $(R - bv)cos\psi$,$(R - bv)sin\psi$ and $M - c\dot{\psi}$ respectively.

$$\begin{aligned}\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} - \lambda a_1 = Q_x \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{y}} - \frac{\partial L}{\partial y} - \lambda a_2 = Q_y \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{\psi}} - \frac{\partial L}{\partial \psi} - \lambda a_3 = Q_\psi\end{aligned} \tag{4}$$

The coefficients to the Lagrange multipliers are the same as the coefficients of the states in the constraint equations.Substituting all the quantities and simplifying we get.

$$m\ddot{x} + \lambda sin\psi = (R - bv)cos\psi \tag{5}$$

$$m\ddot{y} - \lambda cos\psi = (R - bv)sin\psi \tag{6}$$

$$J\ddot{\psi} = M - c\dot{\psi} \tag{7}$$

From Newton's second law, it can easily be seen that the lateral frictional force acting on the skate is the same as the Lagrangian multiplier λ.

$$m\ddot{x}cos\psi + m\ddot{y}sin\psi = R - bv \tag{8}$$

$$-m\ddot{x}sin\psi + m\ddot{y}cos\psi = \lambda \tag{9}$$

By differentiating the constraint equation (2) we get

$$-\ddot{x}sin\psi - \dot{x}\dot{\psi}cos\psi + \ddot{y}cos\psi - \dot{y}\dot{\psi}sin\psi = 0 \tag{10}$$

Therefore

$$m\dot{\psi}(\dot{x}cos\psi + \dot{y}sin\psi) = \lambda \tag{11}$$

Here $\dot{x}cos\psi + \dot{y}sin\psi$ is the longitudinal velocity of the knife edge. Denoting the longitudinal velocity by v, we get

$$m\dot{\psi}v = \lambda \tag{12}$$

Using the expression for longitudinal velocity, (8) can be simplified as

$$m\dot{v} = R - bv \tag{13}$$

. We can combine all these equations into a simple state space form as follows

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\psi}\\ \dot{v}\\ \dot{\omega}\end{bmatrix} = \begin{bmatrix}vcos(\psi)\\ vsin(\psi)\\ \omega\\ \frac{(R-bv)}{m}\\ \frac{(M-c\dot{\psi})}{J}\end{bmatrix} \tag{14}$$

The nonholonomic constraint that the robot cannot move in the direction perpendicular to its orientation given by (2) is satisfied at every instant and can be easily verified by substituting (14) in (2). To simplify the computation process by approximating the infinite dimensional optimization problem to a finite dimensional optimization problem, the dynamics will be discretized as follows and will be used to model the behavior of the robot in the rest of the paper. The discretized equations are obtained using Euler discretization with a sampling interval Δt. A more complicated yet better approximation can be obtained using first order hold or any other similar higher order interpolations for the inputs between the sampling points. The discretized dynamics are given by

$$\begin{bmatrix}x_{t+1}\\ y_{t+1}\\ \psi_{t+1}\\ v_{t+1}\\ \omega_{t+1}\end{bmatrix} = \begin{bmatrix}x_t + v_t cos(\psi_t)\Delta t\\ y_t + v_t sin(\psi_t)\Delta t\\ \psi_t + \omega_t\Delta t\\ v_t + \frac{(R-bv_t)\Delta t}{m}\\ \omega_t + \frac{(M-c\omega_t)\Delta t}{J}\end{bmatrix} \tag{15}$$

An equivalent representation of (15) as following will be used in the Adaptive MPC algorithm.

$$\begin{bmatrix}x_{t+1}\\ y_{t+1}\\ \psi_{t+1}\\ v_{t+1}\\ \omega_{t+1}\end{bmatrix} = \begin{bmatrix}x_t + v_t cos(\psi_t)\Delta t\\ y_t + v_t sin(\psi_t)\Delta t\\ \psi_t + \omega_t\Delta t\\ \alpha_v v_t + \beta_v R\\ \alpha_\omega \omega_k + \beta_\omega M\end{bmatrix} \tag{16}$$

Where

$$\begin{aligned}\alpha_v &= 1 - \frac{b\Delta t}{m} \\ \beta_v &= \frac{\Delta t}{m} \\ \alpha_\omega &= 1 - \frac{c\Delta t}{J} \\ \beta_\omega &= \frac{\Delta t}{J}\end{aligned} \tag{17}$$

It can be easily seen that by treating $\theta = [\alpha_v, \beta_v, \alpha_\omega, \beta_\omega]^T$ as proxy system parameters, the original system parameters $[m, b, J, c]^T$can be calculated using a simple one-to-one transformation. This transformation (17) makes the system dynamics a linear function of the system parameter. This property will become handy to use existing algorithms during the identification of the system parameters. These dynamics can be compactly written as

$$X_{t+1} = f(X_t, u_t, \theta) \tag{18}$$

where $X_k \in \mathcal{X} \subseteq \mathbb{R}^5$ is the state vector and $u_k \in \mathcal{U} \subseteq \mathbb{R}^2$ is the vector of control inputs $[R, M]^T$.The equations in (15) will be used to describe the dynamics of the robots.

Fig. 1. A sample image of a wheeled mobile robot. source: google

III. MPC Algorithm

To begin, we built a constrained finite time optimal control function in Python using the solver "ipopt" from the IPOPT optimization package. This constrained finite time optimal control (CFTOC) function incorporated both state constraints based on system dynamics and input constraints based on the predicted mechanical limits of the robot system. In addition, it was designed to optimize the robot trajectory so that it would arrive at a specific set of desired terminal states. Actual system parameters were given to the function, and a batch approach was used to calculate an optimal set of states and inputs. The results, with all system parameters known, can be seen in figures seven through nine in the Appendix.

This CTFOC function is feasible since Python was able to calculate an optimal series of states and inputs.

To further optimize, the CTFOC function was implemented in a larger MPC function, so that the optimal states and inputs would be recalculated at each time step. Graphs and simulations show the accuracy of the MPC control. However, determining the persistent feasibility of this MPC simulation is difficult to prove since the problem is nonlinear.

Initially the system parameters such as mass, moment of inertia and damping, were assumed to be constant in these simulations. However, to increase the usefulness and relevance of the control algorithm, it was instead assumed that these parameters were unknown, and a new function was implemented to help the controller adapt. See section IV.

IV. Adaptive MPC

In the dynamic model derived in the previous section, we start with no knowledge of the system parameters like the mass, mass moment of inertia and the drag coefficients of the robot and estimate them as we go along with the knowledge of the states of the system with a parameter adaptation algorithm and feed it as parameters to an optimization problem at every instance to obtain the optimal control inputs. We assume that we have full access to the states of the robot.
A typical MPC problem is formulated as follows

$$\begin{aligned} U_t^*(x(t)) = \underset{U_t}{\text{argmin}} & \sum_{k=0}^{N-1} q(X_{t+k}, u_{t+k}) \\ \text{subject to } & X_t = X(t) && (c1) \\ & X_{t+k+1} = f(X_{t+k}, u_{t+k}, \theta) && (c2) \\ & u_{t+k} \in U && (c3) \\ & U_t = \{u_t, u_{t+1}, \ldots, u_{t+N-1}\} && (c4) \end{aligned} \tag{19}$$

where the constraint (c1) is the measurement, (c2) is the system model as a function of parameters, (c3) is the admissible input set and (c4) is the set of optimization variables. The control input at t would be the first component of $U_t^*(x(t))$ $i.e.$, u_t^*.It is worth noting that the constraints on the state are not included here as the system parameters are not known and guaranteeing the states to be confined to a desired set might not be possible. Instead one could use soft constraints for state constraints
MPC algorithm in (19) is not implementable just by itself as the system parameters are unknown. So, we start with a guess of the system parameters and calculate the optimal control corresponding to the guess and update the guess at every instance using a parameter adaptation algorithm. MPC and parameter adaptation run simultaneously as shown in 2.The adaptive MPC structure is formulated as follows with just a minor modification where θ is replaced by $\hat{\theta}_t$ which is an output of the parameter adaptation algorithm.

$$\begin{aligned} U_t^*(x(t)) = \underset{U_t}{\text{argmin}} & \sum_{k=0}^{N-1} q(X_{t+k}, u_{t+k}) \\ \text{subject to } & X_t = X(t) \\ & X_{t+k+1} = f(X_{t+k}, u_{t+k}, \hat{\theta}_t) \\ & u_{t+k} \in U \\ & U_t = \{u_t, u_{t+1}, \ldots, u_{t+N-1}\} \end{aligned} \tag{20}$$

Parameter Adaptation Algorithm (PAA):

A simple series parallel parameter adaptation algorithm from existing literature will be used in tandem with MPC to update the parameters, but any other PAA could be used depending on the complexity of the problem. The series parallel structure for PAA is formulated in this subsection. Due to the transformation shown in (17), dynamical equations containing the system parameters can be written as an inner product of the system parameters and a regressor vector containing functions of the states.

$$\begin{aligned} v_{t+1} &= \theta_v^T \phi_{v,t} \\ \omega_{t+1} &= \theta_\omega^T \phi_{\omega,t} \end{aligned} \tag{21}$$

https://www.overleaf.com/project/5fcfecd8285efd5b453b0860

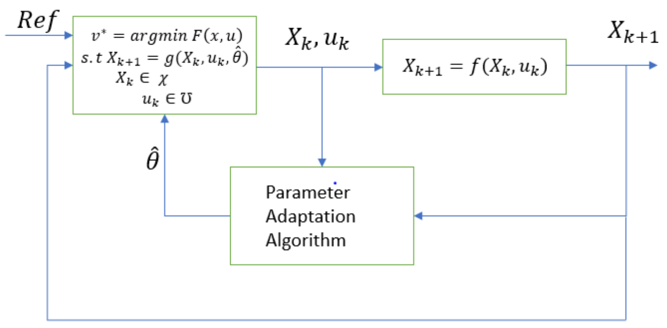

Fig. 2. Adaptive MPC Algorithm

where $\theta_v = [\alpha_v, \beta_v]^T$ and $\theta_\omega = [\alpha_\omega, \beta_\omega]^T$ At every instance with the estimate of the system parameters available, an estimate of the states at the next instance can be found using

$$\begin{aligned}\hat{v}_{t+1} &= \hat{\theta}_{v,t}^T \phi_{v,t} \\ \hat{\omega}_{t+1} &= \hat{\theta}_{\omega,t}^T \phi_{\omega,t}\end{aligned} \tag{22}$$

The error between the actual states measured and the projected states at the next instance can now be used to update the parameters using a series parallel PAA as

$$\begin{aligned}\hat{\theta}_{v,t+1} &= \hat{\theta}_{v,t} + F_{v,t+1}\phi_{v,t}\epsilon_{v,t+1}^0 \\ F_{v,t+1} &= F_{v,t} - \frac{F_{v,t}\phi_{v,t}\phi_{v,t}^T F_{v,t}^T}{1+\phi_{v,t}^T F_{v,t}\phi_{v,t}} \\ \hat{\theta}_{\omega,t+1} &= \hat{\theta}_{\omega,t} + F_{\omega,t+1}\phi_{v,t}\epsilon_{\omega,t+1}^0 \\ F_{\omega,t+1} &= F_{\omega,t} - \frac{F_{\omega,t}\phi_{\omega,t}\phi_{\omega,t}^T F_{\omega,t}^T}{1+\phi_{\omega,t}^T F_{\omega,t}\phi_{\omega,t}}\end{aligned} \tag{23}$$

where $\epsilon_v^0(t+1) = v_{t+1} - \hat{v}_{t+1}$ and $\epsilon_\omega^0(t+1) = \omega_{t+1} - \hat{\omega}_{t+1}$ are the estimation errors and $\hat{\theta}(t) = [\hat{\theta}_v(t)^T \hat{\theta}_\omega(t)^T]^T$. These updated system parameters at every instance can now be used in the Adaptive MPC structure in (20).

V. Numerical Experiments

In this section, the algorithm developed in the previous section will be tested on a mobile robot with mass m = 5 Kg, drag coefficient corresponding to linear motion b = 0.1, Moment of inertia J = 0.2 and drag coefficient corresponding to angular motion c = 0.1. A quadratic cost with all the matrices P,Q and R as identity matrices of proper dimensions is considered. The total horizon M = 500 and a prediction horizon N = 30 is considered to evaluate the algorithm.

$$\begin{aligned}U_t^*(x(t)) = \underset{U_t}{\text{argmin}} \frac{1}{2}e_N^T P_N e_N &+ \sum_{k=1}^{N} \frac{1}{2}e_k^T Q e_k + \frac{1}{2}u_k^T R u_k \\ \text{subject to } X_t &= X(t) \\ X_{t+k+1} &= f(X_{t+k}, u_{t+k}, \hat{\theta}_t) \\ u_{t+k} &\in U \\ U_t &= \{u_t, u_{t+1}, \dots, u_{t+N-1}\}\end{aligned} \tag{24}$$

Performance

As shown in the Fig. 7, using the Adaptive MPC controller, the robot was able to successfully tracked the desired location without too much deviation. Testing different system parameters showed that the robot was able to adapt and continue to successfully follow the desired trajectory, however as system parameters such as mass and moment of inertia became very large, there was more inaccuracy. This may be because of the short distance that we were testing over, as it would be more difficult for a robot with a higher mass moment of inertia to transition the desired heading angle. The robot approached, but did not reach the desired end location and showed more variation in the path it took. That being said, as you can see in Fig. 3, 4 and 5, that as it iterated over parameters it began to approach the true values, thus approaching the trajectory of the system controlled just by MPC.

Conclusion and Future Work

In this work, adaptive MPC was shown to be an effective strategy for optimal control of a simple wheeled robot with unknown parameters. Utilizing a nonlinear system model and introducing an adaptive element to the MPC to control for unknown system parameters makes it more difficult to show persistent feasibility. However, our simulation results look reasonably accurate and have very interesting applications.

There are numerous ways to expand on the work done here. The controller needs to be refined to better guarantee feasibility of adaptive MPC control. Potentially this could be achieved by softening unnecessarily hard constraints. Once a consistently feasible control strategy is implemented, the complexity of the problem could be increased by including additional challenges for the robot such as attempting to track more difficult trajectories or responding to static and dynamic obstacles. And, even further down the line, an implementation of learning MPC would be intriguing, especially if paired with Adaptive MPC.

Attached

Appendix

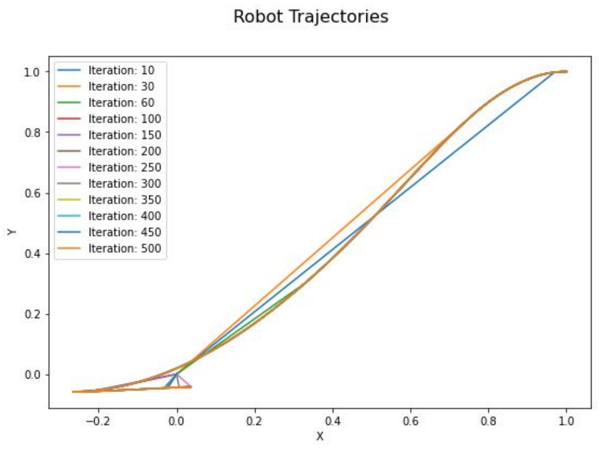

Fig. 3. Robot trajectory while iterating through system parameters

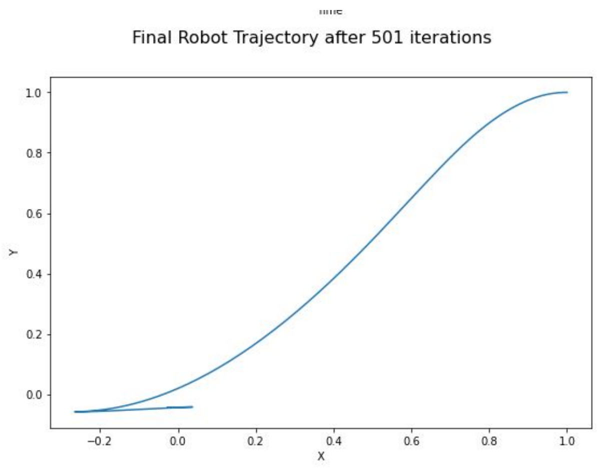

Fig. 4. Final robot trajectory after 501 iterations through parameter adaptation

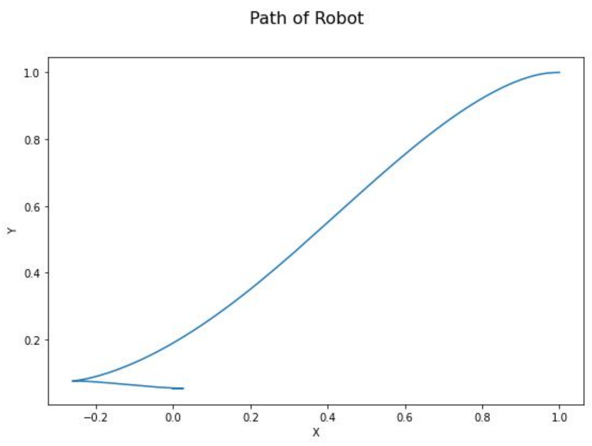

Fig. 7. Path of the robot with MPC with full knowledge of the system

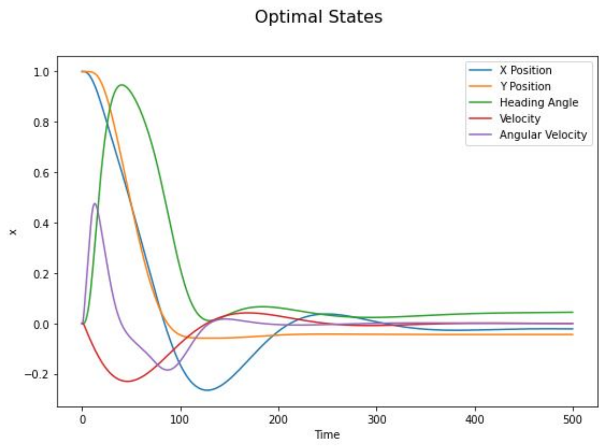

Fig. 5. [Optimal states of the robot with Adaptive MPC

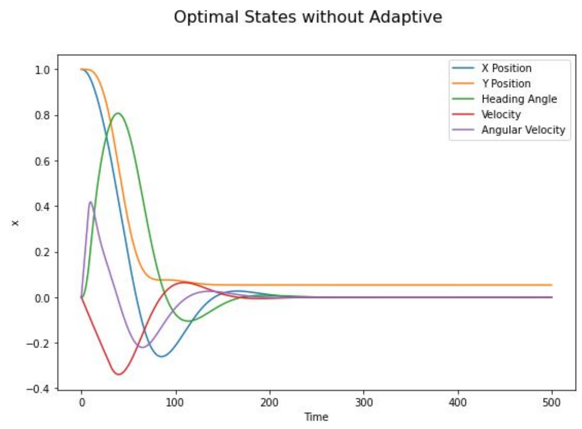

Fig. 8. Path of the robot with MPC with full knowledge of the system

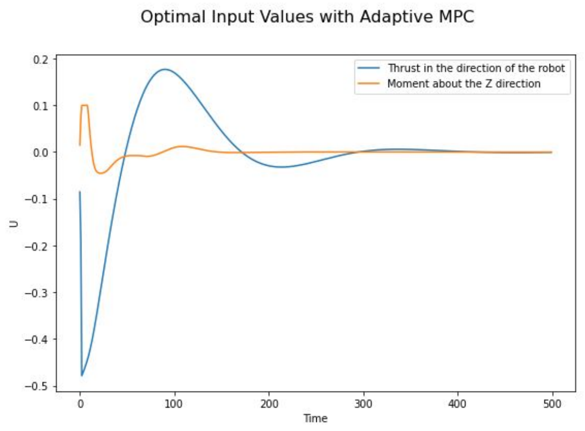

Fig. 6. Optimal input values with no knowledge of the system parameters

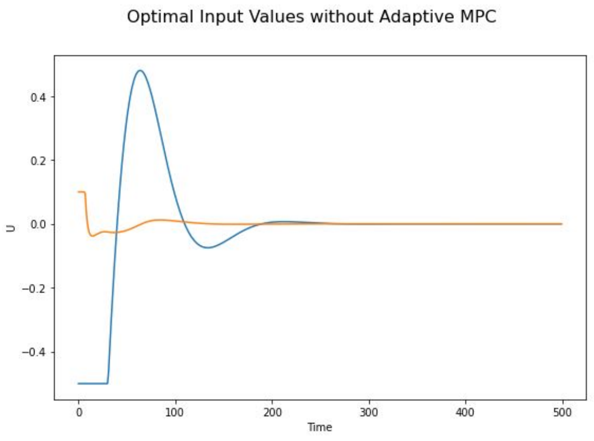

Fig. 9.

References

[1] Felipe Kühne, Walter Fetter Lages, João Manoel Gomes de Silva Jr. “Model Predictive Control of a Mobile Robot Using Linearization.” Federal University of Rio Grande do Sul. Electrical Engineering Department.

[2] Chung-Han Hsieh and Jing-Sin Liu. “Nonlinear Model Predictive Control for Wheeled Mobile Robot in Dynamic Environment” The 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics July 11-14, 2012, Kaohsiung, Taiwan

[3] Ricardo Carona, A. Pedro Aguiar, José Gaspar. “Control of Unicycle Type Robots. Tracking, Path Following and Point Stabilization.” Instituto de Sistemas e Robotica, Instituto Superior Tecnico, Universidade Tecnica de Lisboa, Portugal.

[4] Filippo Bertoncelli, Fabio Ruggiero, Lorenzo Sabattini (2020). Linear Time-Varying MPC for Nonprehensile Object Manipulation with a Nonholonomic Mobile Robot. arXiv:2003.10247v1 [cs.RO]

[5] Marcos, R. O. A. Maximo. “Model Predictive Controller for Trajectory Tracking by Differential Drive Robot with Actuation Constraints.” Instituto Tecnológica de Aeronáutica.