Wheeled Mobile Robot Trajectory

Zhaoqi Li, Chengli Liu

Abstract— **More and more researchers have carried out deep study on wheeled mobile robots because of their wide range of application value. Having a complex environment and being vulnerable to external interference and control constraints, it is difficult to track the trajectory. To solve this problem, linear and nonlinear MPC controller are introduced by using the tool Yalmip and CasADi. Then different constraints, weighting matrices and prediction horizon are used to study the system performance. In addition, disturbance are also considered. Finally a comparison between both controller are drawn.**

I. INTRODUCTION

Nowadays, due to the environment deterioration the oil-consuming cars are slowly being replaced, we pay much attention to the electric car and kinds of mobile robot. In the late 1960s[1], the first mobile robot Shakey was developed, and till now, there has been significant research on the development of wheeled mobile robot, which have found important use in military, civilian, industrial and medical applications. There are mainly two control problem of them, Path tracking and posture stabilization.

They often use its own sensors to complete the tasks given by the system. Since the complexity of the environment and the unaccessible areas the wheeled mobile robot could be considered as a simple-operation, stable-movement and high energy utilization tool to help people dealing with the trivial work. In view of the facts, it is going to be much accounted of the significance for studying and exploring the wheeled mobile robots. In the project we are paying more attention to the research of trajectory tracking.

Based on Model predictive control(MPC), there are several research on the wheeled mobile robots. The research [2] shows a friction compensation method to deal with an omnidirectonal mobile robot in order to realize the path following and control problem. The research [3] represents a robust model predictive control based on neuro-dynamics optimization to follow the trajectory. The previous researches are in the ideal environment, therefore, when the system are uncertain, those MPC would not guarantee the stability of the system. In order to solve these problems, there are many methods, such as, min-max MPC[4], feedback MPC and tube MPC[5].

Wheeled mobile robots have many classifications, from the point of view of number of wheels, they can be classified into two, three, four and six wheels, while according to the wheel drives, they can be considered as differential drive, tricycle, omnidirectional, synchro drive, Ackerman steering, and skid steering.

II. CASE STUDY DESCRIPTION

In this project we use the Ackerman steering model(in the Fig.1) within the wheeled mobile robot in order to follow a referenced car and a designed trajectory, meanwhile, reduce the energy consumption as much as possible. The Ackermann geometry is aimed to avoid the need for tires to slip sideways when following the path around a curve. Though it is complex, this arrangement enhances controllability by avoiding large inputs from road surface variations being applied to the end of a long lever arm, as well as greatly reducing the fore-and-aft travel of the steered wheels.[6]

The simulation oriented model(SOM) is shown in the following figure:

Fig. 1. Ackerman Configuration

This ackerman configuration can be described with a model in the Fig.2

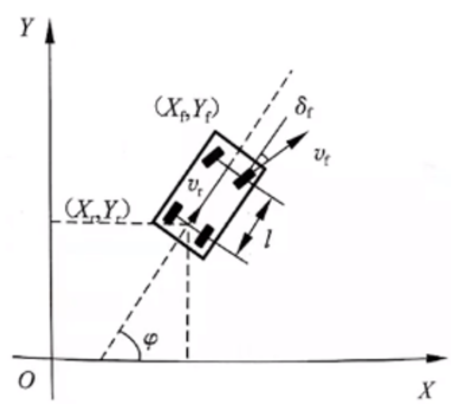

Fig. 2. Ackerman steering model(Simulation oriented model)

Ackerman model is a good first order approximation to vehicle motion, however, given the control inputs predictive control, it is sufficient to consider an even simpler model—a bicycle model—where the left and right wheels are combined into a virtual bicycle, with one front wheel for steering and one rear wheel for driving at the center of the vehicle's longitudinal axis. Hence, in order to simplify the problem we are going to consider the control-oriented model - bicycle model shown in the Fig.3.

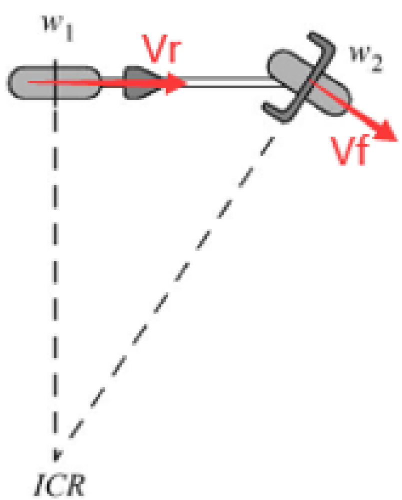

Fig. 3. Bicycle model (Control-oriented model)

The study case can be described as the following:

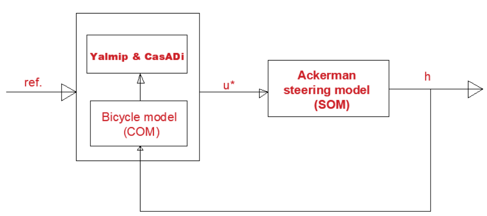

Fig. 4. Setup

III. PROBLEM STATEMENT

In the linear MPC problem it is aimed to follow the referenced robot and we consider two inputs: the steering angle δ_f and the speed of the robot v_r, and three states: position x, y and yaw angle φ in the Cartesian coordinates. The initial states are (12 12 0) and the referenced speed is supposed to be 8 m/s and the referenced steering angle is 0.1 rad.

States 'h' and input 'u':

$$h = \begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} \quad u = \begin{bmatrix} v_r \\ \delta_f \end{bmatrix}$$

As shown in the Fig.3, the bicycle model could be presented with the model shown in the equation (1).

$$\begin{aligned} \frac{dx}{dt} &= v_x \cos\varphi \\ \frac{dy}{dt} &= v_x \sin\varphi \\ \frac{d\varphi}{dt} &= \frac{v_x \tan\delta_f}{l} \end{aligned} \tag{1}$$

With respect to this model we consider its constraints. For the input, the velocity of the robot cannot be too fast because of the secure factor. While in the consideration of the real case the steering angle of the robot should be in the certain range.

There also exist several disturbances, for example, the excessive steering angle change, Wheel sliding, Inaccuracies of the model,etc.

In the Non-linear MPC problem it is aimed to follow a certain designed path and at the same time minimize the energy consumption. In this case we add the dynamic part to see the performance of the system. The new model is shown in the Fig.5.

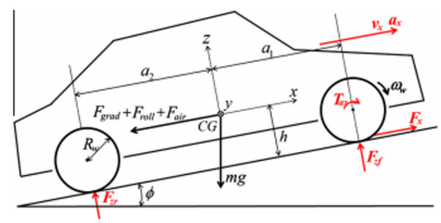

Fig. 5. Dynamic model

We consider two inputs: Electro-mechanical torque T_m and the steering angle δ_f . And six states which are the position x, y , yaw angle φ in the Cartesian coordinates,speed of the next step $v_x - next$, speed change rate δ_{vx}, and power P [7] are considered.

States 'h' and input 'u':

$$h = \begin{bmatrix} x \\ y \\ \varphi \\ v_{x\_next} \\ \delta v_x \\ P \end{bmatrix} \quad u = \begin{bmatrix} T_m \\ \delta_f \end{bmatrix}$$

The speed of the car is no longer an input, it is a intermediate variable derived from the acceleration. According to the figure 5, the total driving forces in the tractive direction, considering the three driving resistances[8], can be expressed as in the equation (2):

$$\sum F_x = F_x - \left(F_{\text{grad}} + F_{\text{roll}} + F_{\text{air}}\right) = F_x - \left(mg\sin\phi + \mu_x mg + k_{\text{air}}\, v_x^2\right) \tag{2}$$

So we could get acceleration a_x shown in the equation (3) from the equation (2), and then we could obtain the speed of the car with integral.

$$a_x = \left(\frac{G_{\text{total}}\,\eta_{\text{total}}}{R_w} T_m - mg\sin\phi - \mu_x mg - k_{\text{air}}\, v_x^2\right)\frac{1}{m+\Delta m} \tag{3}$$

where:

$k_{\text{air}} = 0.5 C_D \rho A$

$\Delta m = \frac{mJ_{\text{total}}\, G_{\text{total}}}{m{R_w}^2 - J_w}$

Parameters:

- Coefficients of rolling resistance μ_x

- Aerodynamic drag C_D
- Air density ρ
- Front projection area of the vehicle A
- Equivalent rotational inertia J_w
- Gear ratio G

$$\begin{gathered} \frac{dx}{dt} = v_x \cos\varphi \\ \frac{dy}{dt} = v_x \sin\varphi \\ \frac{d\varphi}{dt} = \frac{v_x \tan\delta_f}{l} \\ dv_{rx-next} = a_x \cdot dt \\ \Delta v = v_{x-next} - v_x \\ P = \frac{\delta_a G}{2 t_a g}\left(V_f^2 + V_b^2\right) + \frac{2}{3} G f_f V_f + \frac{1}{5} k_e S V_f^2 . \end{gathered} \tag{4}$$

Therefore, the control-oriented model can be described in the equation(4).

In the nonlinear case several uncertainties are going to be considered: the excessive steering angle changeδ_f, variation of the electro-mechanical torque T_m, different kind of pavement μ_x and variation of the massive of the mobile robot m.

IV. MPC CONTROLLER DESIGN

A. Linear MPC Controller Design

Different steps are used to deal with the linear MPC controller design.

Linearization:

Firstly it is going to linearize the model by using the Taylor expansion around the reference.

$$\dot{\tilde{h}} = A(t)\tilde{h} + B(t)\widetilde{u}$$

Discretization:

Then Euler's Method is used to discretize the continuous model.

$$\begin{gathered} \dot{\widetilde{h}} = \frac{\widetilde{h}(k+1) - \widetilde{h}(k)}{T} = A(t)\widetilde{h}(k) + B\widetilde{u}(k) \\ \widetilde{h}(k+1) = A(k)\tilde{h}(k) + B(k)\tilde{u}(k) \\ A(k) = \begin{bmatrix} 1 & 0 & -T v_{r_r}\cos\varphi_r \\ 0 & 1 & T v_{r_r}\cos\varphi_r \\ 0 & 0 & 1 \end{bmatrix} \\ B(k) = \begin{bmatrix} T\cos\varphi_r & 0 \\ T\sin\varphi_r & 0 \\ T\frac{\tan\delta_f}{l} & \frac{T v_r}{l\cos^2\delta_f} \end{bmatrix} \end{gathered}$$

Combination:

Since we are going to encounter a quadratic programming problem, the step of combination is used in order to obtain the feasible results. We combine states and input:

$$\xi(k) = \begin{bmatrix} \tilde{h}(k) \\ \tilde{u}(k-1) \end{bmatrix} \quad \tilde{A} = \begin{bmatrix} A(k) & B(k) \\ O_{m\times n} & \mathrm{Im} \end{bmatrix}$$

$$\widetilde{B} = \begin{bmatrix} B \\ \mathrm{Im} \end{bmatrix} \quad \widetilde{C} = \begin{bmatrix} C & 0 \end{bmatrix}$$

$$\begin{gathered} \widetilde{h}(k) = h(k) - h_r(k) \\ \widetilde{u}(k-1) = u(k-1) - u_r(k-1) \\ \Delta u(k) = \tilde{u}(k) - \widetilde{u}(k-1) \end{gathered}$$

So we could obtain:

$$\begin{gathered} \xi(k+1) = \widetilde{A}\xi(k) + \widetilde{B}\Delta u(k) \\ \eta(k) = \widetilde{C}\xi(k) \end{gathered}$$

Prediction:

$$\begin{gathered} \xi(k+1) = \widetilde{A}\xi(k) + \widetilde{B}\Delta u(k) \\ \xi(k+2) = \widetilde{A}^2\xi(k) + \widetilde{A}\widetilde{B}\Delta u(k) + \widetilde{B}\Delta u(k+1) \\ \xi(k+Hp) = \widetilde{A}^{Hp}\xi(k) + \widetilde{A}^{Hp-1}\widetilde{B}^1\Delta u(k) + \ldots + \widetilde{A}^0\widetilde{B}\Delta u(k+Hs) \end{gathered}$$

So we could obtain:

$$Y(k) = \psi\xi(k) + \theta\Delta u(k)$$

Optimization:

Cost function:

$$J(k) = \sum_{i=1}^{Hp} \|\xi(k+i \mid k)\|_{Q_i}^2 + \sum_{i=0}^{Hp-1} \|u(k+i \mid k)\|_{R_i}^2,$$

$$\min_{\mathbf{u}(k)} J(k)$$

Subject to:

$$\begin{aligned} & \xi(k+i+1) = f(h(k+i), u(k+i)) \\ & \forall i \in [0, Hp-1] \\ & \underline{\xi} \leq \xi(k+i \mid k) \leq \bar{\xi}, \quad \forall i \in [1, Hp] \\ & \underline{u} \leq u(k+i \mid k) \leq \bar{u}, \quad \forall i \in [0, Hp-1] \end{aligned}$$

Parameters:

- $Q = diag(5,5,1,0.1,0.1)$
- $R = diag(1,1)$
- Simulation Horizon: 2 minutes
- Prediction Horizon: 0.3 seconds
- Sampling period: 0.1s
- $\underline{\xi}_1 = -inf.$ and $\overline{\xi}_1 = +inf.$
- $\underline{\xi}_2 = -inf.$ and $\overline{\xi}_2 = +inf.$
- $\underline{\xi}_3 = -180^\circ$ and $\overline{\xi}_3 = 180^\circ$
- $\underline{\xi}_4 = -10m/s$ and $\overline{\xi}_4 = 10m/s$
- $\underline{\xi}_5 = -35^\circ$ and $\overline{\xi}_5 = 35^\circ$
- $\underline{u}_1 = -10m/s$ and $\overline{u}_1 = 10m/s$
- $\underline{u}_2 = -0.6rad$ and $\overline{u}_2 = 0.6rad$

In this case we pay more attention to the robot's position so we penalize much on the variable x and y. From this part we are going to obtain the value of Δu. Then we use the first value of Δu. Finally we do iterations around the previous steps. Finally we obtained the results by using Yalmip.

B. Results of linear MPC design

As is shown in the figure 6, the real velocity of the robot is very similar to the reference one in every steps and at the end it almost converge to 8m/s. And about the steering angle, though at the beginning there exist a relatively bigger deviation, around the 80th step the steering angle of the robot almost catch the objective. Owing to the direction changing, the value of the curve from 0.1rad to -0.1 rad in the middle.

With respect to the state errors(Fig.7), when the robot starts up, there exist errors in both of position and orientation

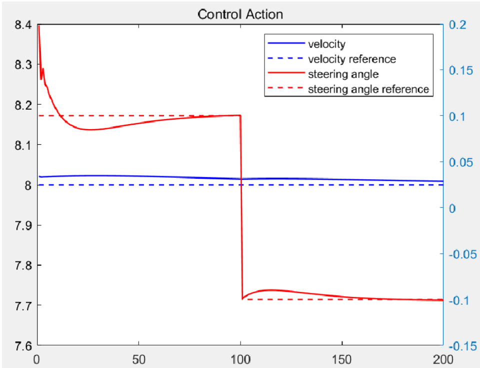

Fig. 6. Control Action

and the position errors are higher than the angle's error. As time goes by, the errors are going to be 0 except in direction-x.

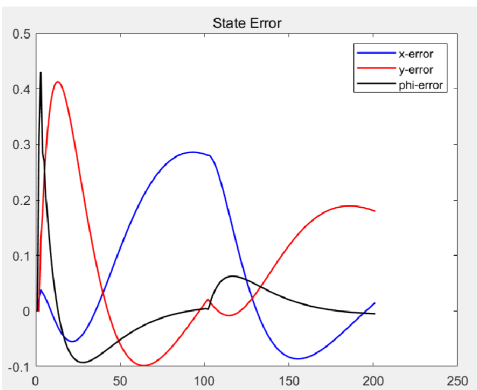

Fig. 7. State errors

And finally the motion simulation are represented in the Fig.8 which show a good overlap between the real robot and the objective.

C. Nonlinear MPC Controller Design

Firstly, it is going to sample the value of continuous time model in time instant "k" and run the integration routine from "k" to "k+1", and then get the data of evolution of continuous model in "k+1".

In this case we penalize the six states and two inputs quadratically respectively, the cost function and constraints considered are shown in the following:

Cost function:

$$J(k) = \sum_{i=1}^{Hp} \|h(k+i \mid k)\|^2_{Q_i} + \sum_{i=0}^{Hp-1} \|u(k+i \mid k)\|^2_{R_i},$$

$$\min_{\mathbf{u}(k)} J(k)$$

Subject to:

$$h(k+i+1) = f(h(k+i), u(k+i))$$
$$\forall i \in [0, Hp-1]$$
$$\underline{h} \leq h(k+i \mid k) \leq \bar{h}, \quad \forall i \in [1, Hp]$$
$$\underline{u} \leq u(k+i \mid k) \leq \bar{u}, \quad \forall i \in [0, Hp-1]$$

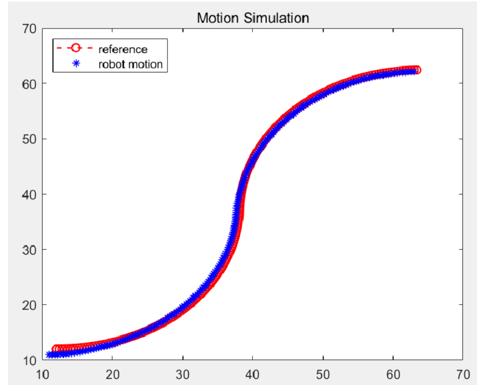

Fig. 8. Motion Simulation

Parameters:

- $Q = diag(1000, 1000, 1, 10, 10, 0.000001)$
- $R = diag(0.5, 0.01)$
- Simulation Horizon: 40 seconds
- Prediction Horizon: 0.6 seconds
- Sampling period: 0.1s
- $\underline{h}_1 = -inf.$ and $\overline{h}_1 = +inf.$
- $\underline{h}_2 = -inf.$ and $\overline{h}_2 = +inf.$
- $\underline{h}_3 = -180°$ and $\overline{h}_3 = 180°$
- $\underline{h}_4 = -10m/s$ and $\overline{h}_4 = 10m/s$
- $\underline{h}_5 = -0.3m/s$ and $\overline{h}_5 = 0.3m/s$
- $\underline{h}_6 = -160kW$ and $\overline{h}_6 = 160kW$
- $\underline{u}_1 = -8NM$ and $\overline{u}_1 = 8NM$
- $\underline{u}_2 = -30°$ and $\overline{u}_2 = 30°$

Concerning the weighting matrix Q, since we are paying more attention to the robot's position, the larger factor will be put with regard to the position. In addition, in consideration of the velocity, we need to minimize the factor of Power. Because with the power reference equal to 0 when the factor is bigger, the power is converging to zero, which causing a small torque and then affecting on the force, thus producing a very small starting velocity. In order to reduce the deviation between the robot's velocity and the reference we put relative big value in the 4th and the 5th item. And there are not many constraints about the yaw angle, so we put the value 1 in the third item. In terms of the weighting matrix R, there would not be more limitations on the torque, thus the factor is going to be small.

When choosing the prediction horizon, it could not be select too long because of the uncertainties of the trajectory nor too short due to the insufficiency of the prediction.

Then we obtained the results by using CasADi.

D. Results of Nonlinear MPC design

Though there exist small errors in the position, the trajectory following results fits well the objective that shown in the Fig.9.

In the Fig.10, the steering angle are changing between -0.5 rad and 0.5 rad and finally it converges around 0. Because the torque is constant from around the 10 time steps, the speed change rate is zero. There is not a much power consumption

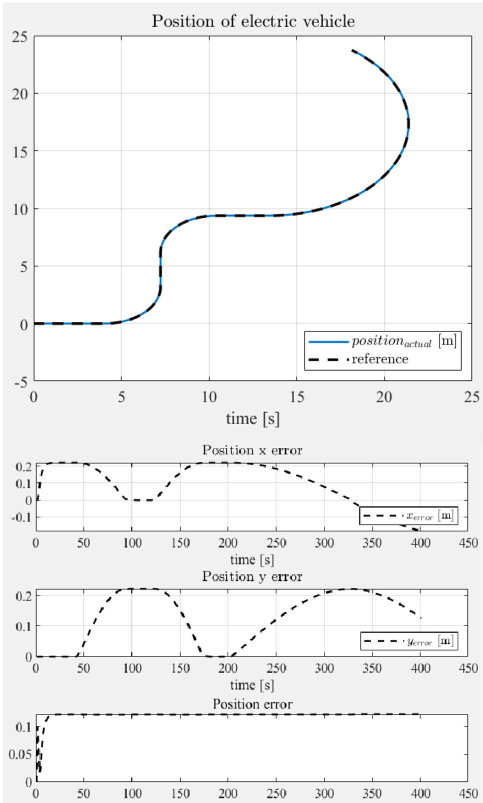

Fig. 9. Motion simulation and Position error

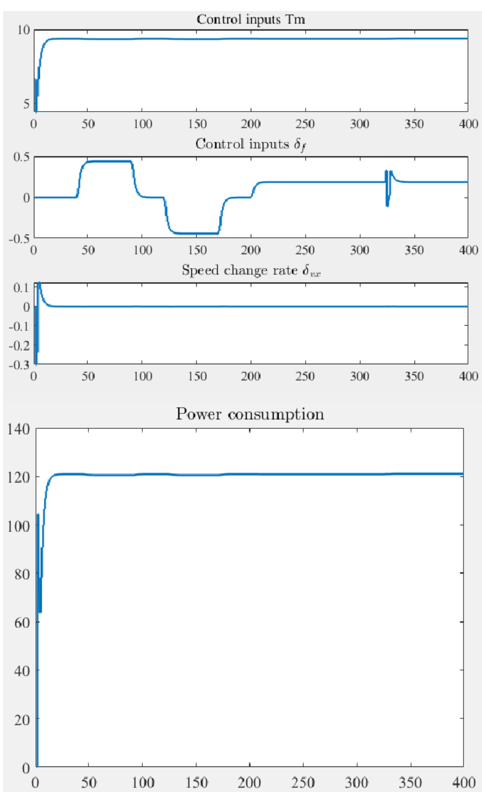

Fig. 10. Simulation results

in the case. In terms of the energy consumption, due to the startup of the robot at the beginning which produces an acceleration, the curve shows a peak in the first few time steps and then reach around 120kW.

V. VARIABLE CHANGING

A. Constraints

Now it is going to do some changes in the constraints. When relax them the results are very similar to the Fig.9 and Fig.10. So we are going to narrow the constraints. In this case, we tried to narrow the range of the power which from [-160, 160]kW to [-80,80]kW. The results are represented in the Fig.11.and Fig.12

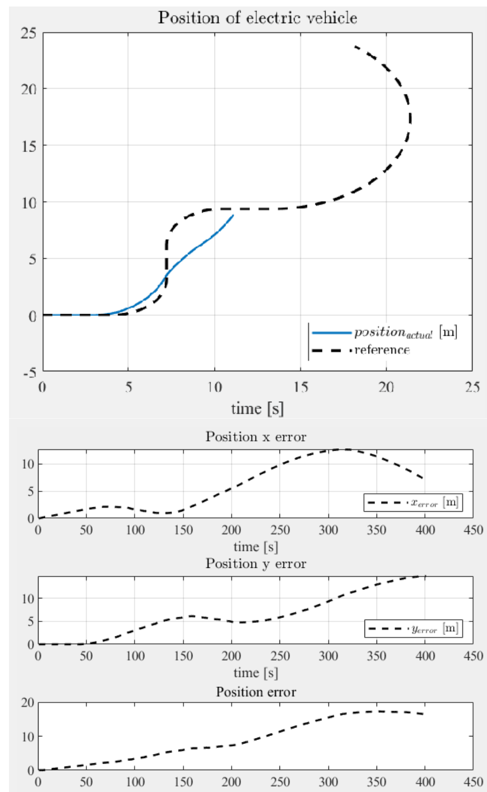

Fig. 11. Motion simulation and Position error

As we can see in the Fig.11, the robot cannot follow the designed path at the end and the position errors are much higher than the previous situation. And as is shown in the Fig.12, when the power is decreasing, the torque will decrease and thus the velocity of the robots will decrease. In every time steps the robot are going to track the objective so several adjustments need to be done, therefore the curve shows a lot of oscillations.

B. Prediction Horizon

In terms of the prediction horizon, we are going to minimize it from 0.6s to 0.4s in order to observe the performance of the system. The results are shown from the Fig.13. to Fig.15.

When shrinking the prediction horizon the robot could also follow the reference trajectory and all of the value keep

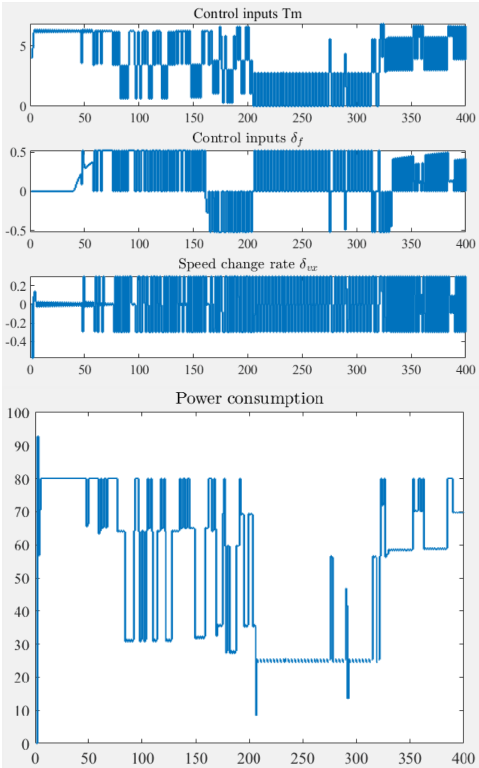

Fig. 12. Simulation results

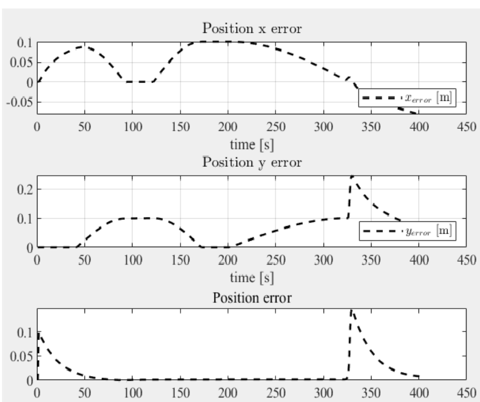

Fig. 13. Simulation results

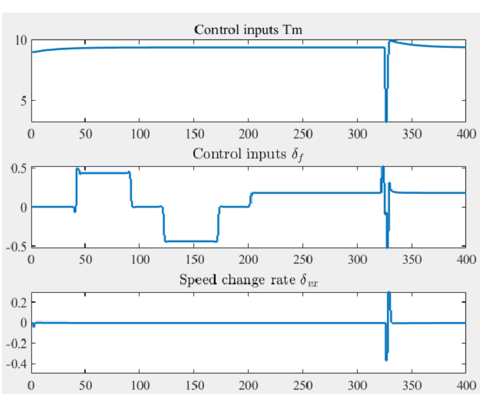

Fig. 14. Simulation results

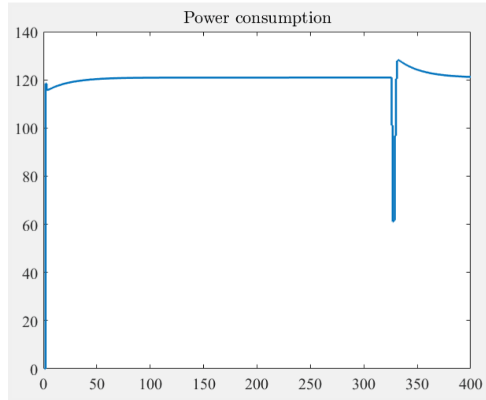

Fig. 15. Simulation results

almost the same but appears a peak in the curve, because there is a huge deviation between the prediction and the real case, in order to adjust this situation the robot deviate the original track at this time step.

C. Weighting Matrix

With regard to the weighting matrix, we tried to change some of the value to see what would happen for the system.

In the first case we change the weights of two positions, the weighting matrices are $Q = diag(100, 100, 1, 10, 10, 0.000001)$ and $R = diag(0.5, 0.01)$. The results are shown in the Fig.16.

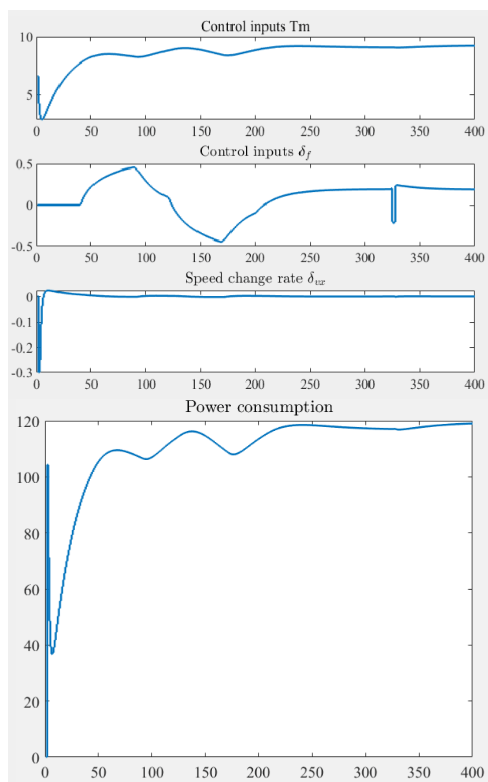

Fig. 16. Simulation results

Since the weights of the position are reduced which means we put less attention to its position, there is a less overlap between the two curves which represent less accuracy for tracking problem. The position errors still exist but there are not too much because we did not decrease much the weights.

And then we select another weight about the power, the weighting matrices are $Q = diag(1000, 1000, 1, 10, 10, 0.1)$ and $R = diag(0.5, 0.01)$

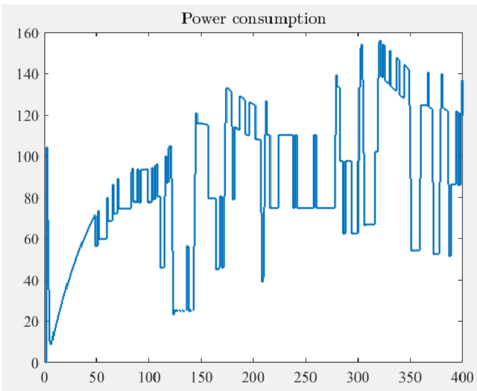

Fig. 17. Simulation results

When increasing the weight of the power, the trajectory did not follow the designed trajectory.In addition, the torque and the steering angle were changing at every time steps. The results are similar to those when narrow the constraints of power, but show the bigger energy consumption in the Fig.17.

D. Disturbance

As is shown in the Fig.18, these exist some small oscillation along the curve but the robot can track the objective well with the small position error and the ideal energy consumption. We could use Kalman Filter to reject the disturbance.

VI. CONCLUSION

We have designed two kinds of MPC controller and according to the obtained results it can be found that linear MPC can be used with drawbacks due to the static errors far from the linearization point. In our case study, Non linear MPC achieves better results than the linear one and processes faster than the linear one.

REFERENCES

[1] Mohamed A. Kamel and Youmin Zhang, Developments and Challenges in Wheeled Mobile Robot Control. 2014 [Online]

[2] Barreto S. Conceicao A G S, Dorea C E T, et al. Design and implementation of model predictive control with friction compensation on an omnidirectional mobile robot[J]. IEEE/ASME Transaction on Mechatronics 2014, 19(2):467-476.

[3] Xiao H, Li Z, Yang C, et al. Robust stabilization of a wheeled mobilr robot using model predictive control based on neuro-dynamics optimization[J].IEEE Transaction on Industrial Electronics 2016(1):505-516.

[4] Liu,C, Li H,Gao J,et al. Robust self-triggered min-max model predictive control for discrete-time nonlinear system[J]. Automatica, 2018,89:333-339.

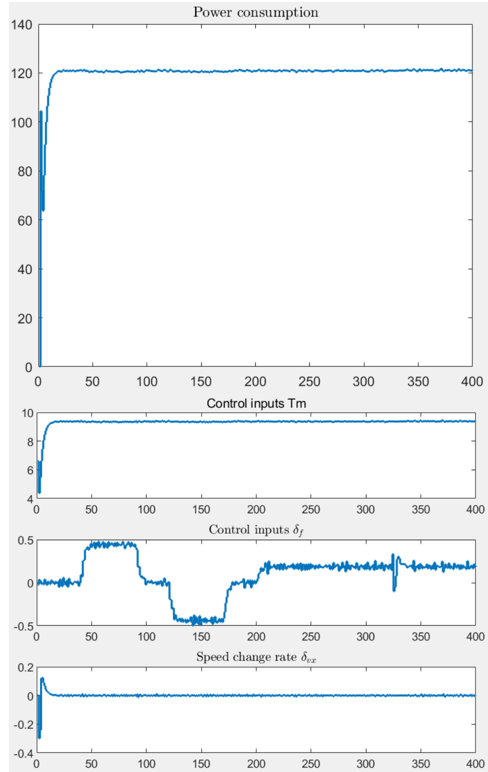

Fig. 18. Simulation results when adding disturbance

[5] Ke K, Li Z, Yang C. Robust tube-based predictive control for visual servoing of constrained differential-drive mobilr robots[J]. IEEE Transaction on Industrial Electronics 2018,65(4):3437-3446.

[6] https://en.wikipedia

[7] Chiara F, Kyoungho A, Hesham A, Power-based electric vehicle energy consumption model: Model development and validation [Online].

[8] Gwangmin P,Seonghun L,Sungho J,Sangshin K, Integrated modeling and analysis of dynamics for electric vehicle powertrains [Online].