2017 21st International Conference on Process Control (PC)
June 6–9, 2017, Štrbské Pleso, Slovakia

Comparitive Study of Predictive Controllers for Trajectory Tracking of Non-holonomic Mobile Robot

Rahul Sharma K.,František Dušek, Daniel Honc
Department of Process Control, Faculty of Electrical Engineering and Informatics
University of Pardubice
nám. Čs. legií 565, 532 10 Pardubice, Czech Republic
rahul.sharma@student.upce.cz, frantisek.dusek@upce.cz, daniel.honc@upce.cz

Abstract—**The paper deals with predictive control of non-holonomic mobile robot. The basic nonlinear kinematic equation is linearized into two different linear time varying models based on frame of reference – world coordinates and local coordinate of mobile robot. The non-linear model predictive control is applied to the trajectory tracking problem of a non-holonomic mobile robot with these models. The control law is derived from a cost function which penalizes the state tracking error, control effort and terminal state deviation error. Various simulation experiments are conducted and a comparative analysis has been made with respect to state-of-the-art approaches.**

Keywords—Trajectory tracking; Optimization; Mobile robot; Predictive control

I. INTRODUCTION

The past few decades have witnessed an increased research effort in the area of motion control of autonomous vehicles. In the age of self-driving cars, the importance of the study of motion control of autonomous systems is ever increasing. Robust motion control algorithms are fundamental to the autonomous operation of mobile robot. Motion control refers to “how to control the robot to make some particular motion – either time bound or not”. There are basically three types of motion control problems: trajectory tracking, path following and point stabilization. Point stabilization (parking) refers to the stabilization of the robot into a predefined position and orientation. Path following refers to move a robot in a path in a time independent manner. The trajectory tracking problem is similar to path following problem, but in predefined time. A typical motion control problem is trajectory-tracking, which is concerned with the design of control laws that force a vehicle to reach and follow, a time parameterized reference (i.e., a geometric path with an associated timing law).

The trajectory tracking problem focuses on stabilizing the robot in a time parameterized reference position and orientation. In doing so non-holonomic constraints must be respected. This means it is not possible to achieve all the velocities at a given moment. According to Brockett’s condition [1], non-holonomic systems cannot be stabilized around equilibrium with smooth time-invariant feedback. However, it has been proven that, the asymptotic stabilization can be obtained using time-varying, discontinuous or hybrid control laws, for e.g. [2-4]. An extensive survey on non-holonomic control problems can be found in [5].

The model predictive control (MPC) (also known as receding horizon control (RHC)) has been an important research area for decades. MPC is also seems to be very promising in the field of mobile robotic trajectory tracking, because the reference trajectory is known beforehand. It is an online optimization tool, which will generate optimal control actions required at every time instant, by minimizing an objective function based on predictions [6], and also by respecting constraints. With the increase in computational power, the MPC is not only limited to slow dynamics processes, but also there are new applications for faster systems. Most of the MPC technologies are based on linear dynamic models and therefore referred to as a linear model predictive controller (LMPC). However, many processes are sufficiently nonlinear which hinder the successful application of LMPC. This has led to the development of nonlinear model predictive controllers (NMPC) in which nonlinear models are used for prediction and optimization. One of the problems associated with NMPC is that, the nonlinear program has to be solved online at every sampling time to generate control action, which is a computationally heavy task. There are various NMPC formulations in the literature, refer [7] and the references within. In case of trajectory tracking of mobile robots, MPC techniques produce promising results as shown by [8-11]. A survey of motion control problems of wheeled mobile robots (WMRs) using MPC can be found in [12].

In this paper, two linear time varying models are derived from the non-linear kinematic model based on the reference coordinate frame. A successive linear model is derived, considering the world coordinates, by successively linearizing around the reference points. An error based linear model is derived, considering the local coordinate of mobile robot, by coordinate transformation. Two trajectory tracking NMPCs are designed with these models, by minimizing a criteria consisting of state deviation error and control effort. Various simulation experiments are carried out and a comparison has been made with respect to state-of-the-art approaches like the Kanayama controller [13] and Samson controller [14]. The paper is organized as follows: section II introduces the kinematic modelling and linearization. Design of NMPC is described in section III and state tracking controllers in section IV. In section V, simulation results of trajectory tracking control and comparison of controllers are provided.

II. Kinematic Model of Non-holonomic Mobile Robot

Let the pose of the robot in the Cartesian coordinate with an angle θ measured clockwise from x-axis, be

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}.$$

The basic kinematic equations of non-holonomic robot are given by,

$$\begin{aligned} \dot{x} &= v\ \cos\theta, \\ \dot{y} &= v\ \sin\theta, \\ \dot{\theta} &= \omega, \end{aligned} \tag{1}$$

where v and ω are the tangential velocity and angular velocity respectively. This can be represented in matrix format,

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}.$$

Trajectory of a mobile robot refers to the locus of all the points (x, y) in the Cartesian coordinate. In a trajectory tracking approach to mobile robot motion control problem, the reference trajectory must be known beforehand. A feasible trajectory considering the velocity and acceleration limits, non-holonomic and holonomic constraints, and an obstacle free trajectory should be generated (by trajectory planner module). The reference trajectory Cartesian coordinates $(x_r\ y_r)$, orientation θ_r, and velocities $(v_r\ \omega_r)$ fulfil the same kinematic equations (1) as,

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}} = \begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \cos\theta_r & 0 \\ \sin\theta_r & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ \omega_r \end{bmatrix}. \tag{2}$$

In a trajectory tracking control of mobile robot, the aim is to minimize the difference between the reference trajectory state vector and the current state vector of mobile robot.

$$\boldsymbol{x}_{\boldsymbol{r}} - \boldsymbol{x} = \begin{bmatrix} x_r \\ y_r \\ \theta_r \end{bmatrix} - \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}.$$

The reference tangential velocity is calculated by,

$$v_r(t) = \sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2}. \tag{3}$$

The reference orientation at every time instant is,

$$\theta_r(t) = arctan2(\dot{y}_r(t), \dot{x}_r(t)). \tag{4}$$

By taking derivative of orientation, the reference angular velocity is obtained as,

$$\omega_r(t) = \dot{\theta}_r(t). \tag{5}$$

The aim of the trajectory tracking controller is to minimize the distance between real and reference robot motion (tracking deviation). There are two major approaches on how to express the tracking deviation, which are further linearized to get an approximate linear model. The starting point of both the models is the basic kinematic equations, but the main difference lies in the choice of the co-ordinate frame of mobile robot and reference trajectory.

A. Successive linear model (M1)

A linear model can be derived from the non-linear model, (1), by successively linearizing around the trajectory of the reference robot. A reference robot can be considered as a robot with reference (desired) parameters of the robot to follow a reference trajectory. The kinematic equations (1) can be represented as a simple model,

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}), \tag{6}$$

where state variables $\boldsymbol{x} = [x, y, \theta]^{\mathrm{T}}$ and control inputs $\boldsymbol{u} = [v, \omega]^{\mathrm{T}}$. Let the reference robot be following a reference trajectory $(x_r\ y_r)$ with an orientation of θ_r. The kinematic equations are the same as that of real mobile robot.

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}} = f(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}}) \tag{7}$$

The reference parameters are $[x_r\ y_r\ \theta_r\ v_r\ \omega_r]$. The tangential velocity, orientation and angular velocity of the reference robot can be calculated from (3-5). Applying the Taylor series approximation to (6), around the time varying reference points $(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}})$, we can derive,

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}}) + \left.\frac{\partial f(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{\substack{x = x_r \\ u = u_r}} (\boldsymbol{x} - \boldsymbol{x}_{\boldsymbol{r}}) + \left.\frac{\partial f(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\substack{x = x_r \\ u = u_r}} (\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{r}})$$

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}}) + \widetilde{\boldsymbol{A}}_S(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}}).(\boldsymbol{x} - \boldsymbol{x}_{\boldsymbol{r}}) + \widetilde{\boldsymbol{B}}_S(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}})(\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{r}}) \tag{8}$$

Subtracting (7) from (8) gives,

$$\Delta\dot{\boldsymbol{x}} = \widetilde{\boldsymbol{A}}_S(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}}).\Delta\boldsymbol{x} + \widetilde{\boldsymbol{B}}_S(\boldsymbol{x}_{\boldsymbol{r}}, \boldsymbol{u}_{\boldsymbol{r}})\Delta\boldsymbol{u} \tag{9}$$

$\Delta\boldsymbol{x}$ is the error vector of state variables and $\Delta\boldsymbol{u}$ is the error vector of control variables with respect to the reference robot. The approximation of $\Delta\dot{\boldsymbol{x}}$ in (9), by the forward differences gives the following discrete-time linear time-variant state-space model:

$$\Delta\boldsymbol{x}(k+1) = \boldsymbol{A}_S(k)\Delta\boldsymbol{x}(k) + \boldsymbol{B}_S(k)\Delta\boldsymbol{u}(k) \tag{10}$$

where,

$$\boldsymbol{A}_S = \begin{bmatrix} 1 & 0 & -v_r \sin\theta_r(k)\, T_s \\ 0 & 1 & v_r \cos\theta_r(k)\, T_s \\ 0 & 0 & 1 \end{bmatrix}, \boldsymbol{B}_S = \begin{bmatrix} \cos\theta_r(k)\, T_s & 0 \\ \sin\theta_r(k)\, T_s & 0 \\ 0 & T_s \end{bmatrix}$$

$$\Delta\boldsymbol{x} = \begin{bmatrix} x(k) - x_r(k) \\ y(k) - y_r(k) \\ \theta(k) - \theta_r(k) \end{bmatrix}, \Delta\boldsymbol{u} = \begin{bmatrix} v(k) - v_r(k) \\ \omega(k) - \omega_r(k) \end{bmatrix}$$

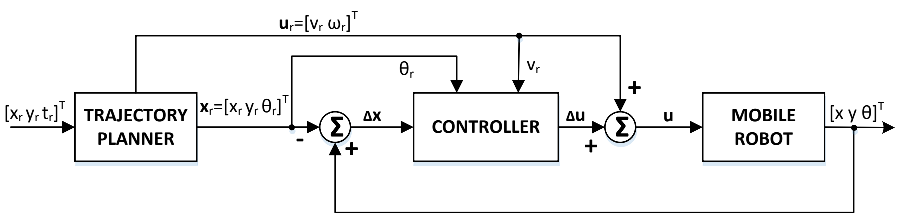

Fig. 1. General block diagram of trajectory tracking kinematic controller with successive linear model

where T_s is the sampling period and $\Delta\boldsymbol{x}$ is the deviation state vector which represents the error with respect to the reference robot, and $\Delta\boldsymbol{u}$ is associated with the control input. The reference values, v_r, θ_r, ω_r are the reference tangential velocity, orientation angle and angular velocity, respectively. A general closed loop control scheme with a feedforward feedback controller is shown in Fig. 1.

B. *Error based linear model (M2)*

Another way of modeling is to consider the difference in the local coordinate system of the mobile robot, see Fig. 3. These differences in the local coordinate system is called the "tracking error" given as,

$$\boldsymbol{e} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} = \boldsymbol{T_x}(\boldsymbol{x_r} - \boldsymbol{x}). \quad (11)$$

where, $\boldsymbol{T_x}$ is the coordinate transformation matrix. Differentiating (11) by considering (1) and (2),

$$\dot{\boldsymbol{e}} = \begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} e_2\omega - v + v_r \cos e_3 \\ -e_1\omega + v_r \sin e_3 \\ \omega_r - \omega \end{bmatrix}. \quad (12)$$

In order to get a linear model, (12) is linearized around equilibrium point ($\boldsymbol{e} = \boldsymbol{0}$), with $[v\ \omega] = [v_r\ \omega_r]$ as the operating points, by the approximating $\sin\theta \approx \theta$ (at small angles – θ is the error variable and the aim is to minimize the errors).

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} 0 & \omega_r & 0 \\ -\omega_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \cos e_3 - v \\ \omega_r - \omega \end{bmatrix} \quad (13)$$

The continuous time state-space model after linearization and approximation is given by,

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & \omega_r & 0 \\ -\omega_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix}}_{\widetilde{A}_E} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} + \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}}_{\widetilde{B}_E} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \quad (14)$$

Separating control inputs as feedforward and feedback inputs,

$$\boldsymbol{u}_{fb} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \underbrace{\begin{bmatrix} v_r \cos e_3 \\ \omega_r \end{bmatrix}}_{\boldsymbol{u}_{ff}} - \underbrace{\begin{bmatrix} v \\ \omega \end{bmatrix}}_{\boldsymbol{u}}. \quad (15)$$

Discretizing (14) with a sample time of T_s, and the discrete time LTV state space model is given by,

$$\boldsymbol{e}(k+1) = \boldsymbol{A}_E(k)\boldsymbol{e}(k) + \boldsymbol{B}_E(k)\boldsymbol{u}_{fb}(k), \quad (16)$$

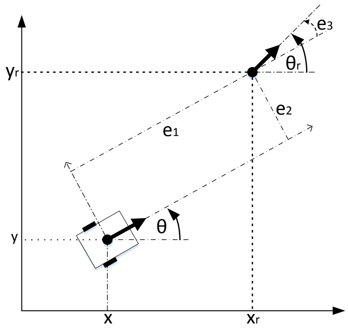

Fig. 3. Coordinate system of real robot and reference robot

where, $\boldsymbol{A}_E$ and $\boldsymbol{B}_E$ are discretized version of matrices $\widetilde{\boldsymbol{A}_E}$ and $\widetilde{\boldsymbol{B}}_E$. A general closed loop control scheme with feedforward feedback controller is shown in the Fig. 2.

III. Trajectory Tracking by NMPC

In this section, the NMPC problem is formulated as a discrete-time optimal control problem with a finite horizon constrained by considering input constraints. At every time instant, the MPC algorithm uses the LTV/non-linear model to predict the evolution of the system for a finite time horizon and generates the optimal control action by optimizing a cost function. Only the first control action of this sequence is applied to the system. The optimization problem is solved again at the next sampling time using the updated process measurements, and a shifted horizon.

The linear models, successive linear and error based model, are LTV models (MIMO system with 2 inputs and 3 outputs). Since state variables correspond to system outputs, the linear discrete time state space model, consists of only the state equation represented as,

$$\bar{\boldsymbol{x}}_{k+1} = \boldsymbol{A}_k \bar{\boldsymbol{x}}_k + \boldsymbol{B}_k \bar{\boldsymbol{u}}_k, \quad (17)$$

where, the state variables, $\bar{\boldsymbol{x}}_k$ and matrices $\boldsymbol{A}_k, \boldsymbol{B}_k$ are same as that in (10) for successive linear model and (16) for error based model.

A. *Prediction model*

The future state variables are recursively calculated from the linear model. Let N be the prediction horizon. The predicted state variables for the finite horizon N at time instant $t_k = kT_s$,

$$\bar{\boldsymbol{X}}_N = \boldsymbol{S}_{k,xx}\bar{\boldsymbol{x}}_k + \boldsymbol{S}_{k,xu}\bar{\boldsymbol{u}}_N, \quad (18)$$

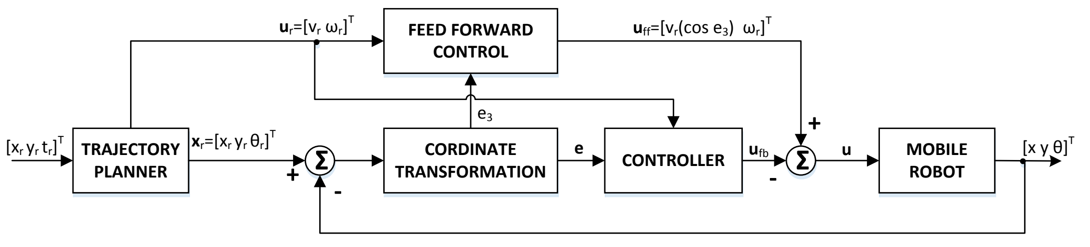

Fig. 2. General block diagram of trajectory tracking kinematic controller with error based linear model

$$\bar{\boldsymbol{u}}_N = \begin{bmatrix} \bar{\boldsymbol{u}}_k \\ \vdots \\ \bar{\boldsymbol{u}}_{k+N-1} \end{bmatrix}; \; \bar{\boldsymbol{X}}_N = \begin{bmatrix} \bar{\boldsymbol{x}}_{k+1} \\ \vdots \\ \bar{\boldsymbol{x}}_{k+N} \end{bmatrix}$$

Decomposing (18) into free $\bar{\boldsymbol{X}}_{N,fr}$ and forced responses $\bar{\boldsymbol{X}}_{N,fo}$,

$$\bar{\boldsymbol{X}}_{N,fr} = \boldsymbol{S}_{k,xx}\bar{\boldsymbol{x}}_k + \boldsymbol{S}_{k,xu}\bar{\boldsymbol{u}}_{N,0}, \tag{19}$$

$$\bar{\boldsymbol{X}}_{N,fo} = \boldsymbol{S}_{k,xu}\Delta\boldsymbol{u}_N. \tag{20}$$

Let $\bar{\boldsymbol{u}}_{fr}$ be the free response control action (last control action) and the optimized control action deviation, $\Delta\boldsymbol{u}_N$, given as,

$$\Delta\boldsymbol{u}_N = \bar{\boldsymbol{u}}_N - \bar{\boldsymbol{u}}_{N,0},$$

$$\bar{\boldsymbol{u}}_{N,0} = \begin{bmatrix} \bar{\boldsymbol{u}}_{fr,k} \\ \vdots \\ \bar{\boldsymbol{u}}_{fr,k+N-1} \end{bmatrix},$$

and the time varying prediction matrices are,

$$\boldsymbol{S}_{k,xx} = \begin{bmatrix} \boldsymbol{A}_k \\ \boldsymbol{A}_{k+1}\boldsymbol{A}_k \\ \boldsymbol{A}_{k+2}\boldsymbol{A}_{k+1}\boldsymbol{A}_k \\ \vdots \\ \boldsymbol{A}_{k+N}\,\boldsymbol{A}_{k+N-1}\dots\boldsymbol{A}_{k+1}\boldsymbol{A}_k \end{bmatrix},$$

$$\boldsymbol{S}_{k,xu} = \begin{bmatrix} \boldsymbol{B}_k & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{A}_{k+1}\boldsymbol{B}_k & \boldsymbol{B}_{k+1} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{A}_{k+2}\boldsymbol{A}_{k+1}\boldsymbol{B}_k & \boldsymbol{A}_{k+2}\boldsymbol{B}_{k+1} & \cdots & \boldsymbol{0} & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \prod_{i=k+1}^{k+N-1}\boldsymbol{A}_i\,\boldsymbol{B}_k & \prod_{i=k+2}^{k+N-2}\boldsymbol{A}_i\,\boldsymbol{B}_{k+1} & \cdots & \boldsymbol{A}_{k+N}\boldsymbol{B}_{k+N-1} & \boldsymbol{B}_{k+N} \end{bmatrix}.$$

B. *Cost function*

The MPC allows a lot of flexibility in the choice of cost function. A general cost function consists of three parts: costs to control error during the horizon, costs to penalize the control signal during horizon and the terminal cost to ensure stability of the control at the terminal state. In the case of the trajectory tracking problem, a separate terminal cost in the criteria formulation is omitted, as the output of system directly corresponds to the state variables.

$$\begin{gathered} J\left(N, \bar{\boldsymbol{x}}_0, \bar{\boldsymbol{u}}_{N,0}\right) = \bar{\boldsymbol{X}}_N{}^T\boldsymbol{Q}\bar{\boldsymbol{X}}_N + \Delta\boldsymbol{u}_N{}^T\boldsymbol{R}\Delta\boldsymbol{u}_N \\ \Delta\boldsymbol{u}_N = \bar{\boldsymbol{u}}_N - \bar{\boldsymbol{u}}_{N,0} \\ \boldsymbol{u}_{min,N} < \boldsymbol{u}_N < \boldsymbol{u}_{max,N} \end{gathered} \tag{21}$$

where the weighting matrix $\boldsymbol{Q}$ is positive semi definite ($\boldsymbol{Q} \succcurlyeq \boldsymbol{0}$) and matrix $\boldsymbol{R}$ is positive definite ($\boldsymbol{R} \succ \boldsymbol{0}$).

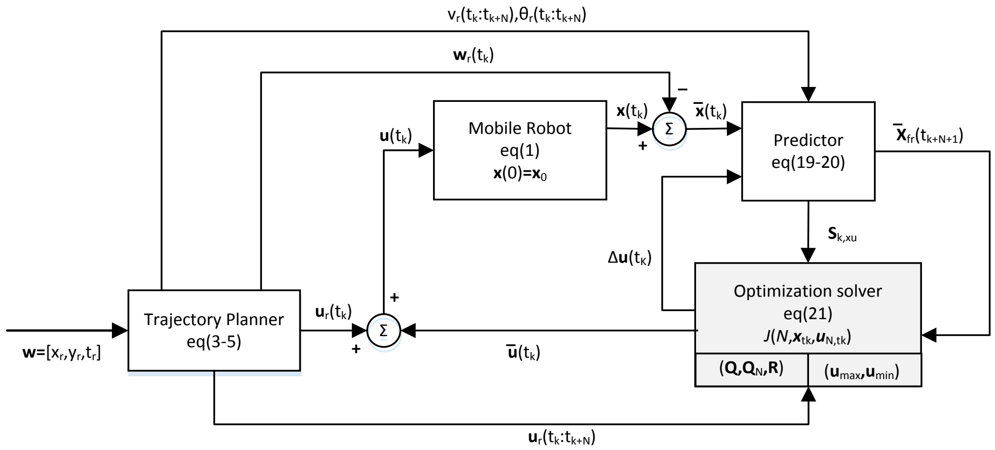

Fig. 4. Control scheme of trajectory tracking NMPC$_1$ with LTV model

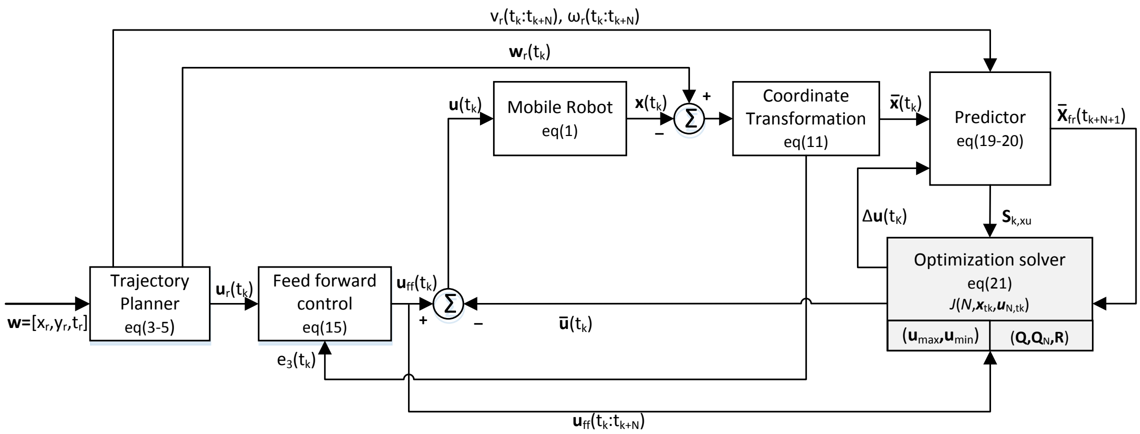

Fig. 5. Control scheme of trajectory tracking NMPC$_2$ with LTV model

$$\left.\begin{array}{l}\boldsymbol{Q} = diag(\boldsymbol{Q_i}) \\ \boldsymbol{R} = diag(\boldsymbol{R_i})\end{array}\right\} \forall\, i = 1 \; to \; N$$

The criteria consist of a cost for control effort, and a cost for state variable deviation. The aim of a trajectory tracking controller is to generate optimum control actions which bring the deviation state variables to zero over a finite time horizon. The last three diagonal elements $\boldsymbol{Q}_N$ in the matrix $\boldsymbol{Q}$, can be seen as terminal state cost, and can be tuned to achieve terminal state stability.

Rewriting the criteria (21), in terms of free and forced response and by substituting (19-20),

$$J = \Delta \boldsymbol{u}_N{}^T \boldsymbol{M} \Delta \boldsymbol{u}_N + \Delta \boldsymbol{u}_N{}^T \boldsymbol{m} + \boldsymbol{m}^T \Delta \boldsymbol{u}_N + \boldsymbol{c} \qquad (22)$$

where,

$$\boldsymbol{m} = \boldsymbol{S}_{k.xu}{}^T \boldsymbol{Q}(\boldsymbol{S}_{k,xx}\bar{\boldsymbol{x}}_k + \boldsymbol{S}_{k,xu}\bar{\boldsymbol{u}}_{N,0})$$
$$\boldsymbol{M} = \boldsymbol{S}_{k,xu}{}^T \boldsymbol{Q} \boldsymbol{S}_{k,xu} + \boldsymbol{R}$$

In case of unconstraint control the analytical solution is,

$$\Delta \boldsymbol{u} = -\boldsymbol{M}^{-1}\boldsymbol{m}. \qquad (23)$$

In case of constraint control, the optimal control action is the solution of the quadratic programing problem, obtained by minimizing the criteria.

$$\min_{\Delta \boldsymbol{u}} J = \Delta \boldsymbol{u}^T M \Delta \boldsymbol{u} + \boldsymbol{2m}^T \Delta \boldsymbol{u} \text{ such that } \boldsymbol{A_o} \Delta \boldsymbol{u} \leq \boldsymbol{b_0}. \qquad (24)$$

Fig. 4 and 5 show the control scheme of NMPC based on successive linear (NMPC_1) and error based models respectively (NMPC_2).

C. Constraints on manipulaed variable

Considering the manipulated variable constraints of a successive linear model for a finite horizon *N*,

$$\begin{gathered}\boldsymbol{u}_{min} \leq \boldsymbol{u}_N \leq \boldsymbol{u}_{max}, \\ \boldsymbol{u}_{min} \leq \Delta \boldsymbol{u}_N + \bar{\boldsymbol{u}}_{N,0} + \boldsymbol{u}_{N,r} \leq \boldsymbol{u}_{max}, \\ \boldsymbol{u}_{min} - \bar{\boldsymbol{u}}_{N,0} - \boldsymbol{u}_{N,r} \leq \Delta \boldsymbol{u}_N \leq \boldsymbol{u}_{max} - \bar{\boldsymbol{u}}_{N,0} - \boldsymbol{u}_{N,r}\end{gathered} \qquad (25)$$

where $\boldsymbol{u}_0$ is the last control action, $\bar{\boldsymbol{u}}_0 = \bar{\boldsymbol{u}}(k-1)$ and $\boldsymbol{u}_{N,r}$ is a vector of reference variables for the horizon. Deriving inequality constraints for a horizon N,

$$\underbrace{\begin{bmatrix} \boldsymbol{I} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{I} & \cdots & \boldsymbol{I} \\ -\boldsymbol{I} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ -\boldsymbol{I} & \cdots & -\boldsymbol{I} \end{bmatrix}}_{A_o} \Delta \boldsymbol{u} \leq \underbrace{\begin{bmatrix} \boldsymbol{u}_{max} - \bar{\boldsymbol{u}}_0 - \boldsymbol{u}_{k,r} \\ \vdots \\ \boldsymbol{u}_{max} - \bar{\boldsymbol{u}}_0 - \boldsymbol{u}_{k+N,r} \\ -\boldsymbol{u}_{min} + \bar{\boldsymbol{u}}_0 + \boldsymbol{u}_{k,r} \\ \vdots \\ -\boldsymbol{u}_{min} + \bar{\boldsymbol{u}}_0 + \boldsymbol{u}_{k+N,r} \end{bmatrix}}_{b_0},$$

and for error based model,

$$\begin{gathered}\boldsymbol{u}_{min} \leq \boldsymbol{u}_N \leq \boldsymbol{u}_{max} \\ \boldsymbol{u}_{min} \leq \boldsymbol{u}_{ff} - (\Delta \boldsymbol{u}_N + \bar{\boldsymbol{u}}_{N,0}) \leq \boldsymbol{u}_{max} \\ \boldsymbol{u}_{min} + \bar{\boldsymbol{u}}_{N,0} - \boldsymbol{u}_{N,ff} \leq -\Delta \boldsymbol{u}_N \leq \boldsymbol{u}_{max} + \bar{\boldsymbol{u}}_{N,0} - \boldsymbol{u}_{N,ff}\end{gathered} \qquad (26)$$

$$\underbrace{\begin{bmatrix} -\boldsymbol{I} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ -\boldsymbol{I} & \cdots & -\boldsymbol{I} \\ \boldsymbol{I} & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{I} & \cdots & \boldsymbol{I} \end{bmatrix}}_{A_o} \Delta \boldsymbol{u} \leq \underbrace{\begin{bmatrix} \boldsymbol{u}_{max} + \boldsymbol{u}_0 - \boldsymbol{u}_{k,ff} \\ \vdots \\ \boldsymbol{u}_{max} + \boldsymbol{u}_0 - \boldsymbol{u}_{k+N,ff} \\ -\boldsymbol{u}_{min} - \boldsymbol{u}_0 + \boldsymbol{u}_{k,ff} \\ \vdots \\ -\boldsymbol{u}_{min} - \boldsymbol{u}_0 + \boldsymbol{u}_{k+N,ff} \end{bmatrix}}_{b_0}$$

IV. State Tracking Controller

A. Linear state tracking control design (Kanayama controller)

The proposed predictive controller is compared to the state of the art state tracking controller whose design can be found in [13-14]. The state tracking controller can be designed with a linear feedback gain as,

$$\bar{\boldsymbol{u}}_k = -\boldsymbol{K}_s(k)\boldsymbol{e}(k) \qquad (27)$$

where $\boldsymbol{K}_s$ is the feedback gain matrix in the form of,

$$\boldsymbol{K}_s(k) = \begin{bmatrix} -k_1(k) & 0 & 0 \\ 0 & -sign\,(v_r)k_2(k) & -k_3(k) \end{bmatrix}. \qquad (28)$$

The controller gains k_1, k_2 and k_3, are determined by comparison (pole placement method) with a desired closed loop characteristic polynomial in the form of,

$$(\lambda + 2\zeta a)(\lambda^2 + 2\zeta a \lambda + a^2)$$

which has constant eigenvalues (one negative real at $-2\zeta a$ and a complex pair with natural frequency $a > 0$ and damping co-efficient $\zeta > 0$). The controller gains can be then chosen as,

$$k_1(k) = k_3(k) = 2\zeta a,$$
$$k_2(k) = \frac{a^2 - \omega_r(k)^2}{v_r(k)}$$

The gain k_2 will go to infinity as $v_r(k) \to 0$. In order to avoid this, a gain scheduling can be designed by letting $a = a(k) = \sqrt{\omega_r(k)^2 + bv_r(k)^2}$, substituting,

$$k_1(k) = k_3(k) = 2\zeta\sqrt{\omega_r(k)^2 + bv_r(k)^2}\ ;\ k_2(k) = b|v_r(k)|,$$

where the factor $b > 0$ can be seen as additional degree of freedom.

Even the controller gains are chosen in such a way that the closed loop poles are at the left half of the s-plane, while the controller is still non-linear and time varying. Therefore, asymptotic stability of tracking error is not guaranteed. The control scheme is same as that in Fig. 2.

B. Nonlinear state tracking control design (Samson controller)

Considering the nonlinear feedback control law [15] as,

$$\boldsymbol{K}_s(k) = \begin{bmatrix} -k_1(k) & 0 & 0 \\ 0 & -\bar{k}_2 v_r(k)\frac{\sin(e_3)}{e_3} & -k_3(k) \end{bmatrix}. \qquad (29)$$

The controller gains $k_1, \bar{k}_2$ and k_3 are determined by the same method as in the linear control design.

$$k_1(k) = k_3(k) = 2\zeta\sqrt{\omega_r(k)^2 + bv_r(k)^2}\ \ ;\ \bar{k}_2 = b.$$

The main difference between a linear and nonlinear state tracking controller is that, global asymptotic stability can be proved in the case of a nonlinear controller by Lyapunov analysis. See [15] for the proof.

V. Simulation Results

The inputs are time parameterized reference points which are interpolated to generate smooth trajectory points by spline

TABLE I. COMPARISON OF TRAJECTORY TRACKING CONTROLLERS

Simulation Experiment	Controller	Tuning parameters				Constraints $u_{max}= -u_{min}$	Initial Condition	Control Quality $[SSE_{xy}\ SSE_{\theta}]$
		N	R	Q	Q_N			
S1	NMPC$_1$	5	10^{-1}*I	I	0	-NA-	Same	[0.0059 2.6030]
S2					I	[1 1]		[0.2668 2.8050]
S3			10^{-2}*I		10^{3}*I	-NA-	[0.1 -0.1 0]	[1.9297 6.8506]
S4	NMPC$_2$	5	10^{-1}*I	I	I	-NA-	Same	[0.0192 0.0004]
S5						[1 1]		[0.0345 2.3777]
S6						-NA-	[0.1 -0.1 0]	[0.1736 57.371]
		b		ζ				
S7	KC	100		0.7		-NA-	Same	[0.0013 0.0067]
S8		50				[1 1]		[0.0101 1.4151]
S9						-NA-	[0.1 -0.1 0]	[0.2837 67.778]
S10	SC	100		0.7		-NA-	Same	[0.0013 0.0067]
S11		50				[1 1]		[0.0101 1.4157]
S12						-NA-	[0.1 -0.1 0]	[0.3188 74.659]

interpolation (MATLAB function `interp1`). The trajectory planner generates the reference parameters – orientation, tangential and angular velocities. Simulation experiments, with a continuous time model (1) for real robot, were performed with a sampling time of 100ms. Total simulation time was 30s. Four different controllers were simulated – NMPC with successive linear model (NMPC$_1$), NMCP with error tracking model (NMPC$_2$), Kanayama feedback controller (KC) and Samson feedback controller (SC).

Trajectory tracking NMPC$_1$ of mobile robot was simulated: by using the model in the form of (10), predicting the future states with LTV model (19-20), optimizing the cost function in the form of (21), and defining the constraints in the form of (25). Optimized control actions for horizon N were calculated and the first control action was applied to the system. Only NMPC with LTV model was considered, as the results obtained with non-linear model, (1), were same. NMPC$_2$ uses error based model, (16), prediction model (19-20) and constraint definition as in (26). The control actions for state feedback tracking controllers KC and SC were calculated by (28) and (29) respectively. The input constraints were considered as,

$$-1\,m/s \leq v \leq 1\,m/s \quad ; \quad -1\,rad/s \leq \omega \leq 1\,rad/s. \quad (30)$$

Twelve different simulation experiments (**S1-S12**) had been performed with different controllers, initial conditions, tuning parameters and constraint condition. Different initial condition refers to the pose of robot, which is different from the reference pose. Fig. 6 and 7 shows the trajectory tracking responses with unconstraint NMPC$_1$ (**S1**) and constraint NMPC$_2$ (**S5**) respectively. Table 1 shows the simulation results of trajectory tracking with different controllers. The results are comparable with the sum of squared error (SSE):

$$SSE_{xy} = |x - x_r| + |y - y_r|\,;\ SSE_{\theta} = |\theta - \theta_r|.$$

Three sets of experiments were conducted – unconstraint control, constraint control and different initial conditions. In all the cases NMPC$_1$ showed more SSEs when compared to other controllers. The SSE of unconstraint control responses

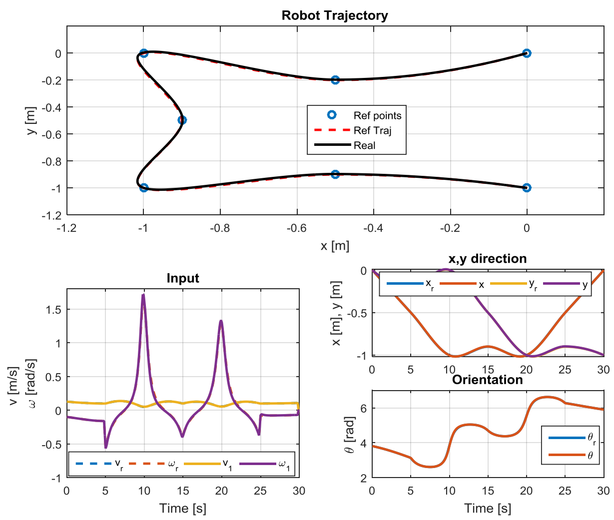

Fig. 6. Trajectory tracking with unconstraint NMPC$_1$ – reference trajectory, reference inputs, tracked trajectory, control actions

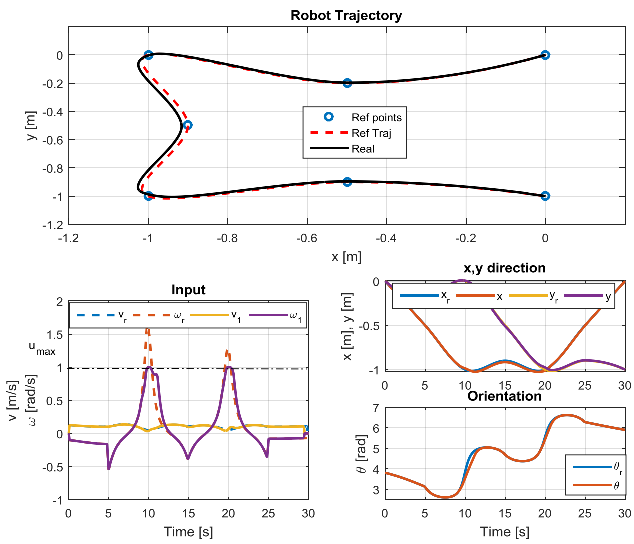

Fig. 7. Trajectory tracking with constraint NMPC$_2$ – reference trajectory, reference inputs, tracked trajectory, control actions

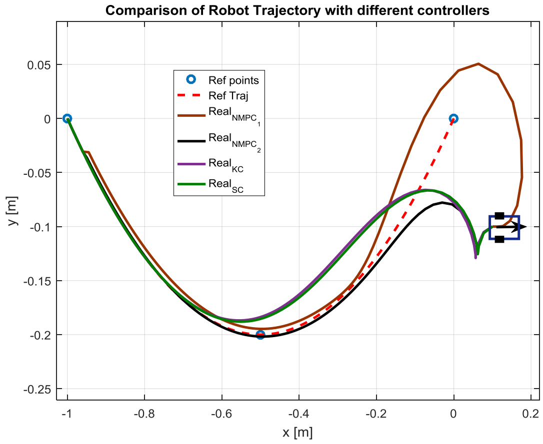

Fig. 8. Comparison of trajectory tracking controllers with different initial conditions

trajectories were almost same, even though $NMPC_1$ is outperformed by all the other controllers. The controllers were able to generate target velocities with respect to the reference velocities. In case of constraint control, state tracking controllers were able to track the robot closer to the reference trajectory. The constraints in the form of (30) were considered. When the initial conditions were different $NMPC_2$ performed better than all the other controllers.

Fig. 8 shows the initial tracking of the mobile robot in the case where the robot's initial pose is different from the reference trajectory – since for this example, the robot is orientated in the opposite direction. $NMPC_2$ converges faster to the reference trajectory compared to all other controllers, followed by state tracking controllers. It is also interesting to note that the $NMPC_1$ initially drives in the opposite direction to the reference orientation and eventually converges with high initial tracking errors.

VI. Conclusion

The basic non-linear kinematic equations of non-holonomic robot is linearized based on two reference coordinate systems. A non-linear MPC is designed with criteria penalizing state tracking errors and control effort. The performance (based on state tracking errors) of trajectory tracking NMPCs were compared with two state-of-the-art approaches. The NMPC based on a local coordinate system showed better results when the initial conditions were different. With same initial conditions and with/without input constraints, all the controllers - other than NMPC based on world coordinates – exhibited similar performances.

Acknowledgment

This research was supported by project SGS, Modern Methods for Simulation, Control and Optimization at FEI, University of Pardubice. This support is very gratefully acknowledged.

References

[1] BROCKETT, RW, "Asmpototic stability and feedback stabilization," Differential Geometric Control Theory, pp. 181-191, 1983.

[2] Kolmanovsky, Ilya, and N. Harris McClamroch, "Developments in nonholonomic control problems," IEEE Control Systems, vol. 15.6, pp. 20-36, 1995.

[3] Sarkar, Nilanjan, Xiaoping Yun, and Vijay Kumar, "Control of mechanical systems with rolling constraints application to dynamic control of mobile robots," The International Journal of Robotics Research, vol. 13.1, 55-69, 1994.

[4] De Luca, Alessandro, and Giuseppe Oriolo. "Modelling and control of nonholonomic mechanical systems". Springer Vienna, 1995.

[5] Oriolo, Giuseppe, Alessandro De Luca, and Marilena Vendittelli, "WMR control via dynamic feedback linearization: design, implementation, and experimental validation," Control Systems Technology, IEEE Transactions on, vol. 10.6, pp. 835-852, 2002.

[6] Camacho, Eduardo F., and Carlos Bordons Alba, Model predictive control, Springer Science & Business Media, 2013.

[7] Henson, Michael A, "Nonlinear model predictive control: current status and future directions," Computers & Chemical Engineering vol. 23.2, pp. 187-202, 1998.

[8] Maurovic, Ivan, Mato Baotic, and Ivan Petrovic, "Explicit model predictive control for trajectory tracking with mobile robots," Advanced Intelligent Mechatronics (AIM), 2011 IEEE/ASME International Conference on. IEEE, 2011.

[9] Sharma, K. Rahul, Daniel Honc and Frantisek Dusek, "Predictive Control Of Differential Drive Mobile Robot Considering Dynamics And Kinematics," ECMS, 2016.

[10] Kuhne, Felipe, Walter Fetter Lages, and Joao Manoel Gomes da Silva Jr, "Model predictive control of a mobile robot using linearization," Proceedings of mechatronics and robotics. 2004.

[11] Lages, Walter Fetter, and Jorge Augusto Vasconcelos Alves, "Real-time control of a mobile robot using linearized model predictive control," Proc. of 4th IFAC Symposium on Mechatronic Systems, 2006.

[12] Kanjanawanishkul, K., "Motion control of a wheeled mobile robot using model predictive control: A survey," KKU Research Journal, vol. 17.5, pp. 811-837, 2012.

[13] Kanayama, Yutaka, Yoshihiko Kimura, Fumio Miyazaki, and Tetsuo Noguchi, "A stable tracking control method for an autonomous mobile robot," In Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on, pp. 384-389, 1990.

[14] De Luca, Alessandro, Giuseppe Oriolo, and Marilena Vendittelli, "Control of wheeled mobile robots: An experimental overview," Ramsete. Springer Berlin Heidelberg, pp. 181-226, 2001.

[15] Samson, Claude, and K. Ait-Abderrahim, "Feedback control of a nonholonomic wheeled cart in cartesian space," Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on. IEEE, 1991.