Control Engineering Practice 78 (2018) 160–174

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

MPC-based control architecture of an autonomous wheelchair for indoor environments

Gianluca Bardaro, Luca Bascetta, Eugenio Ceravolo, Marcello Farina [*], Mauro Gabellone, Matteo Matteucci

Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano Piazza Leonardo da Vinci, 32-20133 Milano, Italy

A R T I C L E I N F O

MSC:
00-01
99-00

Keywords:
Autonomous vehicle
Unicycle (also denoted differential-drive) model
Model predictive control
Ambient assisted living
Autonomous wheelchair

A B S T R A C T

In this paper a linear MPC control scheme is proposed to address the motion problems of an autonomous wheelchair in a realistic environment. Thanks to an inner feedback-linearizing loop, the formulation of the model predictive control problem is simplified, allowing for a real-time computationally-efficient implementation. Thanks to the MPC framework, constraints like obstacle avoidance, actuator limitations, and passenger comfort have been included in the optimization problem. Experimental results show the effectiveness of the proposed scheme.

1. Introduction

In many application domains, autonomous and semi-autonomous vehicles play a fundamental role, e.g., in air (Alexis, Nikolakopoulos, & Tzes, 2012; Dydek, Annaswamy, & Lavretsky, 2013; Ryll, Bülthoff, & Robuffo Giordano, 2015), sea (Cervantes, Yu, Salazar, Chairez, & Lozano, 2016; Millán, Orihuela, Jurado, & Rubio, 2014), land (Hu, Wang, Yan, & Chen, 2016; Wang, Jing, Hu, Yan, & Chen, 2016), and space (Truszkowski, Hinchey, Rash, & Rouff, 2006) explorations, in operations in dangerous and/or unknown environments, in mine detection, in agricultural applications (Liu, Wang, & Zhou, 2008), in the Ambient-Assisted Living (AAL) field (Celeste, Filho, Filho, & Carelli, 2008; Sinyukov & Padir, 2015), and they are also envisaged as promising solutions for enhancing road safety (Cavanini, Benetazzo, Freddi, Longhi, & Monteriù, 2014; Guo, Hu, & Wang, 2016; Lefèvre, Carvalho, & Borrelli, 2016). In particular, control in AAL is crucial to improve the life quality of – especially elderly and physically impaired – people, e.g., in home and hospital environments (Cavanini et al., 2014; He et al., 2017; Leaman & La, 2017; Li et al., 2017; Parikh, Grassi, Kumar, & Okamoto, 2007; Zhang et al., 2016).

Autonomous vehicles are sophisticated systems, endowed with advanced sensing, actuation, processing, and data transmission capabilities. As far as the software is concerned, a large number of algorithms must run in parallel and in real-time (including, e.g., localization, object and obstacle recognition, and a complex hierarchical control structure including a number of different planning and control layers) to provide efficient, reliable, robust, and safe operation. At the core of these complex systems lies a control algorithm, that aims at conferring to the vehicle robust stability and suitable performance levels in different control tasks, e.g., parking, trajectory tracking, obstacle avoidance, etc.

The approaches traditionally used by the robotics community for trajectory tracking and collision avoidance of mobile robots and autonomous vehicles are based on the Dynamic Window Approach (DWA) (Fox, Burgard, & Thrun, 1997) and on the Timed Elastic Band (TEB) (Rösmann, Feiten, Wösch, Hoffmann, & Bertram, 2012, 2013; Rösmann, Hoffmann, & Bertram, 2015) algorithms. DWA is based on the online definition of an optimal control action by sampling the control space, generating feasible paths and choosing the best one between them based on an optimality criteria. TEB, instead, takes the initial trajectory generated by a planning algorithm and performs and online optimization minimizing the trajectory execution time, separation from obstacles and compliance with kinodynamic constraints such as satisfying maximum velocities and accelerations.

Besides usual control tasks, like trajectory tracking or parking, other control specifications, e.g., smooth acceleration and deceleration profiles and collision avoidance requirements, must be properly formulated to guarantee safety and comfort. Model Predictive Control (MPC) (Rawlings & Mayne, 2009) has gained interest, in the past decade, in the

[*] Corresponding author.
E-mail address: marcello.farina@polimi.it (M. Farina).

https://doi.org/10.1016/j.conengprac.2018.06.020
Received 15 December 2017; Received in revised form 28 June 2018; Accepted 30 June 2018

field of autonomous vehicle control, in view of the possibility to cast the motion problems into optimization ones, where operational constraints can be enforced and where predictions can be included. Indeed, MPC allows to develop modular control strategies, where the aforementioned requirements can be formulated in terms of constraints or elements of the cost function, that can even support different operating scenarios, e.g., wheelchair indoor navigation and sidewalk or road driving, by changing on the fly constraints or cost function elements. It is worth noting also that robust MPC implementations are also available, to account for disturbances and model inaccuracies (Rawlings & Mayne, 2009).

In the context of MPC-based autonomous vehicle control, in many works a hierarchical scheme is proposed, consisting of (i) a (event based or slow timescale) local planner that guarantees obstacle avoidance features, possibly based on a rough vehicle mathematical model and (ii) a low-level (path-following) controller, based on a high-fidelity mathematical model. For example, in Falcone, Borrelli, Asgari, Tseng, and Hrovat (2007) this scheme is applied to a 6-states bicycle model with constant wheel angular speed. For numerical complexity reduction, the model is linearized at each time step around the current operating point, leading to a LTV system. Related works (Gao et al., 2012; Gao, Lin, Borrelli, Tseng, & Hrovat, 2010; Gray et al., 2012) propose different solutions for local planning. In Bernardini, Cairano, Bemporad, and Tseng (2009) and Palmieri, Barbarisi, Scala, and Glielmo (2009) dynamic path following layers, similar to the one developed in Falcone et al. (2007), are developed also for lateral dynamics, side-slip control, or yaw rate reference tracking, based on full vehicle models. Also in Katriniok and Abel (2011), a similar low-level stabilizing controller is proposed, to work at the limits of the model dynamics, and where the linearized prediction model proposed in Falcone et al. (2007) is performed, at each time step, not about the current operating point, but about future estimated trajectories, resulting in a time-varying prediction model. A robust low-level MPC implementation is proposed in Bahadorian, Savkovic, Eaton, and Hesketh (2012) using a bicycle model, linearized about the reference trajectory to be tracked. In Raffo, Gomes, Normey-Rico, Kelber, and Becker (2009) a similar hierarchical scheme is proposed: a linear MPC-based local planning (called vehicle guidance) level is devised using the linearized kinematic model of the Ackermann steered vehicle, controlling position and orientation using the desired steering angle as control variable; the dynamic control layer aims to control the yaw rate and the chassis side-slip.

In later works the two levels (i.e., local planner and path follower) are fused together. In Frasch et al. (2013) a nonlinear MPC (NMPC) algorithm solved online with advanced numerical optimization allows to account for obstacle and road traffic constraints, modeled as bounds on the state vector. Here a four-wheel vehicle dynamical model with wheel dynamics and load transfer is used, transformed to a position-dependent one, which is also the approach taken in Plessen, Bernardini, Esen, and Bemporad (2018). In Gao, Gray, Carvalho, Tseng, and Borrelli (2014), a robust tube-based MPC strategy is used considering LTV models, while in Lenz, Kessler, and Knoll (2015) chance constraints are used for comfortable and safe driving. In Ji, Khajepour, Melek, and Huang (2017) and Rasekhipour, Khajepour, Chen, and Litkouhi (2017), the reference trajectory is defined based on a potential field, considering both the goal and the obstacles. General theoretical grounds for trajectory-tracking and path-following controllers are established in Alessandretti, Aguiar, and Jones (2013) and Faulwasser and Findeisen (2016).

Note that distributed MPC-based schemes have also been developed for vehicle formation stabilization (Chen, Sun, Yang, & Chen, 2010; Dunbar & Murray, 2006; Keviczky, Borrelli, Fregene, Godbole, & Balas, 2008), multi-vehicle guidance (Kuwata, Richards, Schouwenaars, & How, 2007), multiple vehicle coordination (Farina, Perizzato, & Scattolini, 2015; Xie & Fierro, 2007).

With specific reference to control of the unicycle-type model, which is the one considered in this paper, two main approaches are taken. For reference trajectory tracking, the most common approach consists of linearizing the tracking error model around the reference trajectory. This approach is taken, e.g., in Kühne, Lages, and Gomes Da Silva (2004) and in Bahadorian et al. (2012) and Gonzalez, Fiacchini, Guzman, and Alamo (2009), where robust MPC is used. On the other hand, the NMPC is adopted for point stabilization (Kühne, Lages, & Gomes Da Silva, 2005; Xie & Fierro, 2008) and for general problems (Gu & Hu, 2006; Maniatopoulos, Panagou, & Kyriakopoulos, 2013). In these papers, collision avoidance constraints are not addressed, but only state convex regions are admitted, including the field-of-view state constraints defined in Maniatopoulos et al. (2013).

In this work we rely on linear models thanks to a standard feedback linearization procedure, similar to the one adopted in Oriolo, De Luca, and Vendittelli (2002) and in Farina et al. (2015); this approach allows to formulate operational, comfort, and collision avoidance requirements as linear constraints: the MPC optimization problem can be thus greatly simplified, allowing for real-time efficient implementation, as witnessed by the experimental results.

Note, however, that in Farina et al. (2015) the proposed control approach is a distributed one, tailored for completing multi-agent tasks (e.g., formation control and coverage), and that it is applied to a team of small-scale educational unicycle robots, with no real realization problems (e.g., localization, real-time implementation in the ROS platform, actuation).

A preliminary control scheme for wheelchair control based on this approach and preliminary results can be found in Ceravolo, Gabellone, Farina, Bascetta, and Matteucci (2017). The present work, however, besides including more thorough discussions, presents the overall control scheme (including localization, planning, and obstacle detection functionalities) in a more rigorous and comprehensive fashion, as well as the implementation/realization choices. Also, it essentially differs from the preliminary paper (Ceravolo et al., 2017) in many respects. For example, the control scheme has been implemented in C++ and embedded, in the form of a local planner, in the Robot Operating System (ROS) (Quigley et al., 2009) standard navigation stack. Also, in this work we have performed a number of new, more comprehensive, and more realistic experimental tests, including a test where two persons are moving in the working area.

The paper is organized as follows. Section 2 describes the model of the system under control, the actuator dynamics, the feedback linearization procedure, and the main regulator structure. Section 3 focuses on the MPC control problem, including the definition of the cost function and of the constraints, while Section 4 introduces some implementation details. Section 5 shows significant experimental tests, while conclusions are drawn in Section 6. Finally, in Appendix, some technical details on the feedback linearization procedure are given.

2. Feedback linearization and control-oriented model of the wheelchair dynamics

In this section we first introduce the nonlinear kinematic model of the wheelchair, showing the results of the actuator identification phase. Moreover, we describe how an internal control loop for the wheelchair is designed, based on the feedback linearization technique, to obtain a versatile discrete-time linear control-oriented model, to be used by a suitable MPC control algorithm.

2.1. Unicycle model

From a kinematic point of view, the motion of a wheelchair can be represented using the *unicycle* nonlinear model, i.e.,

$$\begin{cases} \dot{x}(t) = v_{\text{long}}(t)\ \cos\theta(t) \\ \dot{y}(t) = v_{\text{long}}(t)\ \sin\theta(t) \\ \dot{\theta}(t) = \omega(t) \end{cases} \tag{1}$$

where $x(t)$, $y(t)$ and $\theta(t)$ are the state variables representing the wheel axle center position and orientation in the global reference system (see Fig. 1). The input variables are commonly the longitudinal and angular velocities $v_{\text{long}}(t)$ and $\omega(t)$, respectively.

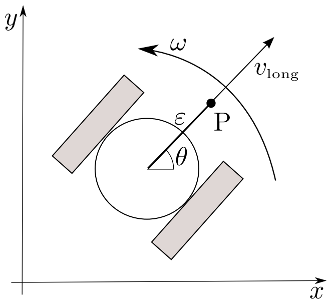


Fig. 1. Unicycle model.

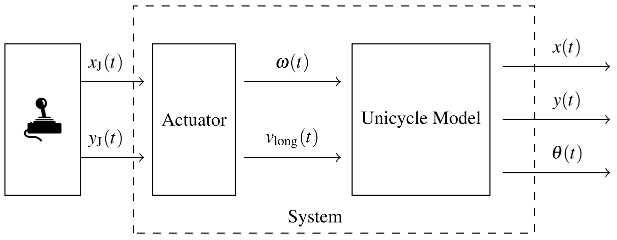


Fig. 2. The actuation system.

2.2. Actuator dynamics

The wheelchair considered in this work (see Section 5 for a detailed description) can be manually controlled by means of an on-board joystick, whose horizontal and vertical displacements in the xy-plane, denoted by $x_J(t)$ and $y_J(t)$, respectively, correspond to velocity reference commands. An internal control loop controls the wheelchair longitudinal and angular velocities in order to meet the setpoints given by the joystick displacements (see Fig. 2).

In this work, it has been therefore considered a reasonable solution to use, as control variables, the joystick positions $x_J(t)$ and $y_J(t)$, which prevented us to redesign the on-board motor controllers, that would not be accepted by the manufacturer. To override the manual control, a suitable communication bus is used to provide the inputs $x_J(t)$ and $y_J(t)$ to the actuator control system from an external computer.

On one hand, thanks to this design choice, the proposed methodology can be easily added as an additional module to existing commercial wheelchairs. On the other hand, the adoption of a cascaded control structure, constituted by the inner wheel velocity loops provided by the wheelchair manufacturer and by the outer MPC tracking controller, allows to simplify the control system design. In particular, the low-level velocity control loops are in charge of nonlinear effects and wheelchair dynamics, as a consequence the MPC layer can be designed considering only a kinematic model of the vehicle.

For the application of a dedicated control scheme, the first step of our work consisted of the identification of the relationships between the joystick position in the xy-plane and the actual velocity inputs $v_{\text{long}}(t)$ and $\omega(t)$. To this aim we have performed a series of open-loop tests by applying step-wise variations to the positions (i.e., the x and y displacements, respectively) of the joystick, see Fig. 3, left panels. This resulted in the trajectories of $v_{\text{long}}(t)$ and $\omega(t)$ shown in Fig. 3, right panels. Note that there is an extremely sharp correlation between the input $x_J(t)$ and the output $\omega(t)$, as well as between $y_J(t)$ and $v_{\text{long}}(t)$. Moreover, no apparent relation is displayed between $y_J(t)$ and $\omega(t)$, as well as between $x_J(t)$ and $v_{\text{long}}(t)$: this can be concluded by observing the plots in Fig. 3. For example from time 50 s to about 150 s $x_J(t)$ displays significant variations, while $v_{\text{long}}(t)$ is essentially constant, like $y_J(t)$. Likewise, from time 0 s to about 50 s $y_J(t)$ displays significant variations, while $\omega(t)$ is constant, similarly to $x_J(t)$. The resulting model is therefore a MIMO decoupled one.

The relationships between $\omega(t)$ and $x_J(t)$ and between $y_J(t)$ and $v_{\text{long}}(t)$ can be well approximated with second-order transfer functions having complex-conjugate poles through standard step response analysis. However, in this work, we have opted to approximate the actuator model as a gain. Indeed, $\omega(t) \simeq \mu_x x_J(t)$ and $v_{\text{long}}(t) \simeq \mu_y y_J(t)$, where $\mu_x = 0.0225$ and $\mu_y = 0.007871$ if $y_J \geq 0$, while $\mu_y = 0.009444$ if $y_J < 0$. This simplification has been done to reduce the difficulties and the risk of singularities, which may occur in the transfer function inversion in real applications (especially in case there is some uncertainty on the transfer functions' parameters). Such approximation can allow to simply generalize the control approach to different wheelchair settings, but also to other similar wheelchair models.

In the MPC framework, the difference between the real actuator model and the static one can be mitigated by properly constraining the speed variation between two consecutive sample times. This approach is appropriately taken in this work (see Section 3.2). Also, possible small-impact model mismatches can be accounted for as non-idealities and perturbations and can be properly counteracted by resorting to standard precautions in MPC practice, like constraint tightening and the use of slack variables. The effectiveness of this approach is demonstrated in the experiments described in Section 5.

2.3. Feedback linearization

An internal static but nonlinear control loop (see Fig. 4) is designed based on the feedback linearization procedure (Megretski, 2003; Oriolo et al., 2002). This is beneficial, since it allows to make the system display a linear dynamics under a suitable change of variables. In turn, the so-obtained linear model is prone to be controlled using a standard MPC scheme, since it allows to formulate quadratic optimization problems to be solved online with limited computational effort.

To this aim, we define a point P placed at distance ε from the wheel axle center in the direction of the longitudinal velocity (see Fig. 1), whose coordinates are

$$\begin{cases} x_P(t) = x(t) + \varepsilon \ \cos\theta(t) \\ y_P(t) = y(t) + \varepsilon \ \sin\theta(t) \end{cases} \tag{2}$$

Differentiating (2) with respect to time and considering (1), we obtain the following linear system

$$\begin{cases} \dot{x}_P(t) = v_{Px}(t) \\ \dot{y}_P(t) = v_{Py}(t) \end{cases}$$

provided that we satisfy the following equality

$$\begin{bmatrix} v_{\text{long}}(t) \\ \omega(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & \sin\theta(t) \\ -\dfrac{1}{\varepsilon}\sin\theta(t) & \dfrac{1}{\varepsilon}\cos\theta(t) \end{bmatrix} \begin{bmatrix} v_{Px}(t) \\ v_{Py}(t) \end{bmatrix} \tag{3}$$

The low-level linearizing layer, including the feedback linearization and the inversion of the actuator dynamics identified in Section 2.2, is sketched in Fig. 4. Some properties on the feedback-linearized models are analyzed in Appendix.

2.4. Discrete time model

Thanks to the feedback-linearization internal control loop designed in the previous section, the discrete-time linear model used for control design is

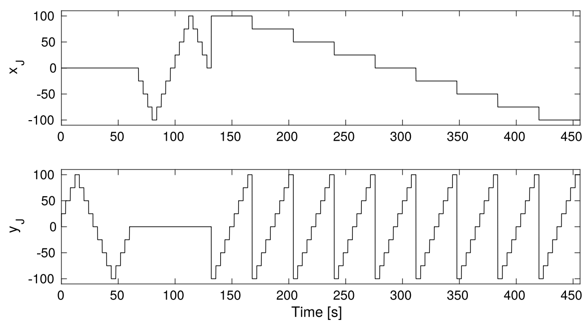
$$\begin{cases} x_P(k+1) = x_P(k) + \tau_s \ v_{Px}(k) \\ y_P(k+1) = y_P(k) + \tau_s \ v_{Py}(k) \end{cases}$$

where τ_s is the controller sampling time. This discrete-time model can be rewritten in a more compact form as

$$\xi(k+1) = \mathbf{A}\ \xi(k) + \mathbf{B}\ \mathbf{u}(k) \tag{4}$$

where $\mathbf{A} = I_2$, $\mathbf{B} = \tau_s I_2$ (being I_2 the 2×2 identity matrix) and

$$\xi(k) = \begin{bmatrix} x_P(k) \\ y_P(k) \end{bmatrix} \qquad \mathbf{u}(k) = \begin{bmatrix} v_{Px}(k) \\ v_{Py}(k) \end{bmatrix}$$


(a) Joystick positions

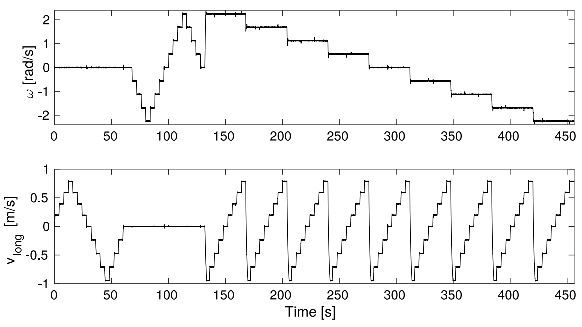

(b) Wheelchair velocities

Fig. 3. Inputs (a) and outputs (b) used in the identification procedure.

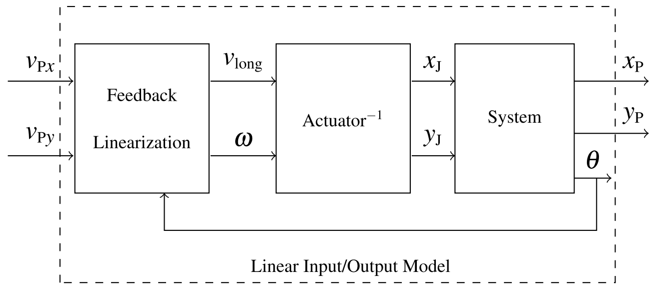


Fig. 4. Feedback linearization internal control loop.

3. MPC control scheme

In this section the design of the MPC controller is illustrated. Specifically, first we focus on the detailed definition of the ingredients of a suitable minimization problem to be solved at each time instant $k \geq 0$. This consists of the definition of cost function and of constraints, including terminal ones, which will be required to ensure recursive feasibility properties (Mayne, Rawlings, Rao, & Scokaert, 2000), at least in nominal conditions. Also, we will resort to suitable slack variables to cope with non-idealities. Finally we will describe how the MPC constrained optimization program is formulated, based on the previously-defined ingredients.

3.1. Cost function

The cost function is formulated based on the main control goals. Specifically, in this work we aim to reduce the error between the state variable $\xi(k)$ evolution and its reference ξ_{ref}, while limiting the energy spent to take the control action $\mathbf{u}(k)$ during a whole future prediction and control horizon, which is N steps long. The cost function results the following.

$$J(k) = \sum_{i=0}^{N-1} \left(\left\| \xi(k+i) - \xi_{ref} \right\|_{\mathbf{Q}}^2 + \|\mathbf{u}(k+i)\|_{\mathbf{R}}^2 \right) + \left\| \xi(k+N) - \xi_{ref} \right\|_{\mathbf{P}}^2 \quad (5)$$

where $\xi(k+i)$ is the i-steps ahead state prediction computed based on the current state $\xi(k)$, on the input sequence $\mathbf{u}(k), \ldots, \mathbf{u}(k+i-1)$, and on the model (4). Matrices $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{P}$ are the weights associated to state and control errors, and to the state error at the end of the control horizon, respectively, conveniently tuned to ensure stability (Rawlings & Mayne, 2009).

The position reference ξ_{ref} commonly corresponds to the desired goal position defined by the user or, as later discussed in Section 4.1, by the global planner. However, when an obstacle is detected between the actual wheelchair position and the goal, a vector field-generated (as proposed, e.g., in Buizza Avanzini, Zanchettin, & Rocco, 2015; De Medio, Nicolò, & Oriolo, 1991) alternative reference may be defined, thus allowing to circumvent the obstacle without incurring into deadlock rest situations. As more detailly described in Section 4, in a few words the alternative vector field-generated reference is basically a point lying on the tangential line with respect to the obstacle perimeter, re-defined online at each time step until the obstacle gets overcome. Note that we allow the definition of different weighting matrices $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{P}$ in the cost function (5) in case the alternative vector field-generated goal is used.

3.2. Constraints

The optimization problem is subject to constraints on the control and state variables; they allow to fulfill physical actuator limitations, physical position constraints, and are required to shape the desired behavior of the system, e.g., based on comfort needs.

Velocity constraints.

When considering differential drive models of type (1), constraints on the velocities of the right and left wheels v_{right} and v_{left}, respectively, must be enforced. To this aim it is sufficient to enforce an upper bound to the speed of point P, i.e., to $\bar{v} = \sqrt{v_{Px}^2 + v_{Py}^2}$. In fact, from (3), we obtain that $\bar{v} = \sqrt{v_{\text{long}}^2 + \varepsilon^2\omega^2}$. Therefore, if we enforce a constraint of the type $\bar{v} \leq v_{\max}$ for a suitable value of $v_{\max}$, then it is guaranteed that both $|v_{\text{long}}| = |v_{\text{right}} + v_{\text{left}}|/2 \leq v_{\max}$ and $\varepsilon|\omega| = |v_{\text{right}} - v_{\text{left}}|\varepsilon/L \leq v_{\max}$, where L is the wheel axle length (in our case $L \simeq 0.55$ m and, as specified in Table 1, $\varepsilon = 0.5$ m). This guarantees $|v_{\text{right}}|$ and $|v_{\text{left}}|$ to remain constrained, as required.

For this reason, in the optimization problem to be solved at the discrete-time instant k, an upper bound on the longitudinal speed of point P is enforced for time k and for all future steps ahead $k+1, \ldots, k+N-1$. Specifically, for $i = 0, \ldots, N-2$ we require that

$$0 \leq \sqrt{v_{Px}^2(k+i) + v_{Py}^2(k+i)} \leq v_{\max} \quad (6)$$

Since constraint (6) is nonlinear, for inclusion in a quadratic and numerically simple optimization problem it is convenient to linearize it. One option, adopted in this paper, is to use first-order truncated Taylor series to linearize (6) around the predicted trajectories $v_{Px}(k+i|k-1)$ and $v_{Py}(k+i|k-1)$, $i = 1, \ldots, N-2$, obtained as solutions to the MPC optimization problem at time instant $k-1$. In this way we reformulate (6) as the following linear bounds

$$0 \leq \frac{v_{Px}(k+i|k-1)}{\bar{v}(k+i|k-1)}\, v_{Px}(k+i) + \frac{v_{Py}(k+i|k-1)}{\bar{v}(k+i|k-1)}\, v_{Py}(k+i) \leq v_{\max} \quad (7)$$

where

$$\bar{v}(k+i|k-1) = \sqrt{v_{Px}(k+i|k-1)^2 + v_{Py}(k+i|k-1)^2}$$

Formally, the constraint that is included in the MPC optimization problem, for each $i = 0, \dots, N-2$, is

$$\begin{bmatrix} \frac{v_{\mathrm{P}x}(k+i|k-1)}{\bar{v}(k+i|k-1)} & \frac{v_{\mathrm{P}y}(k+i|k-1)}{\bar{v}(k+i|k-1)} \\ -\frac{v_{\mathrm{P}x}(k+i|k-1)}{\bar{v}(k+i|k-1)} & -\frac{v_{\mathrm{P}y}(k+i|k-1)}{\bar{v}(k+i|k-1)} \end{bmatrix} \mathbf{u}(k+i) \le \begin{bmatrix} v_{\max} \\ 0 \end{bmatrix} \tag{8}$$

Note that, if $\bar{v}(k+i|k-1)$ is zero or close to zero, numerical issues can compromise the correct inclusion of these constraints in the optimization problem. In this condition, and in particular when $\bar{v} \le v_{\mathrm{low}}$, constraint (6) is replaced by the two decoupled ones: $|v_{\mathrm{P}x}(k+i)| \le v_{\max}/\sqrt{2}$ and $|v_{\mathrm{P}y}(k+i)| \le v_{\max}/\sqrt{2}$. The latter are formally written as follows

$$\begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} \mathbf{u}(k+i) \le v_{\max}/\sqrt{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{9}$$

Note that (9) is more conservative than (6). However, since we adopt (9) only in case of low velocities, this does not compromise the performance of the controller. Finally, it is important to remark that, to account for linearization inaccuracies, in applying (9), the value of $v_{\max}$ should be selected in a slightly conservative way.

Constraints on the velocity variation.

To make the MPC solution consistent with the system dynamics (with reference to the maximum allowed acceleration), but also to conform to passenger comfort (that is also related to the maximum wheelchair acceleration in the x and y directions British-Standard-Institution, 1987; International-Organization-for-Standardization, 1997) and to trajectory smoothness needs, the velocity variation between two consecutive time steps is limited. These constraints are imposed in a decoupled way both in the x and in the y directions. For $i = 1, \dots, N-1$, we require that

$$\begin{aligned} &-\Delta v_{\max} \le v_{\mathrm{P}x}(k+i) - v_{\mathrm{P}x}(k+i-1) \le \Delta v_{\max} \\ &-\Delta v_{\max} \le v_{\mathrm{P}y}(k+i) - v_{\mathrm{P}y}(k+i-1) \le \Delta v_{\max} \end{aligned} \tag{10}$$

where $\Delta v_{\max}$ is the maximum variation for $v_{\mathrm{P}x}$ and $v_{\mathrm{P}y}$. Formally, this corresponds to include, for each $i = 0, \dots, N-1$, the following constraint inequalities to the optimization problem

$$\begin{bmatrix} -1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} \mathbf{u}(k+i-1) \\ \mathbf{u}(k+i) \end{bmatrix} \le \Delta v_{\max} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{11}$$

Note that, for $i = 0$, $\mathbf{u}(k+i-1)$ is known, and so it must not be accounted for as a free optimization variable.

Obstacle avoidance

Obstacle avoidance constraints are also enforced. To this aim, the allowed admissible convex state region according to obstacle positions in the environment must be defined according to a suitable algorithm (see Section 4.3 for further details). This allows to define a set of n_c linear inequalities to be used as additional state constraint in the optimization problem, i.e.,

$$h_1^{(j)}\, x_{\mathrm{P}}(k+i) + h_2^{(j)}\, y_{\mathrm{P}}(k+i) \le l^{o,(j)} \quad j = 1, \dots, n_c \tag{12}$$

shaping the perimeter of the admissible region. Here $h_1^{(j)}$, $h_2^{(j)}$ and $l^{o,(j)}$ are the coefficients representing the sides of the region. Note that constraints (12) must be defined, by suitable tightening, in order to guarantee collision avoidance, not only for point P, but also for the whole wheelchair surface area. This can be done by replacing $l^{o,(j)}$ with $l^{(j)} = l^{o,(j)} - r_w\sqrt{(h_1^{(j)})^2 + (h_2^{(j)})^2}$, where r_w is the radius of the round light gray area depicted in Fig. 5. Note that this approach is not particularly conservative, since the point P lies inside of the general wheelchair area (in our work, it corresponds to the footboard position, see Fig. 9).

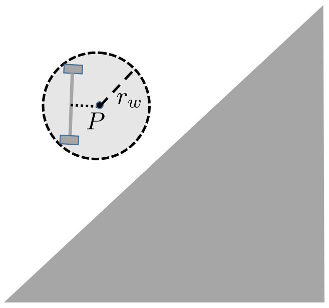


Fig. 5. Linear state constraints.

Overall, (12) is used, in the optimization problem, to define a linear constraint of the type

$$\mathbf{H}\xi(k+i) \le \mathbf{l} \tag{13}$$

where

$$\mathbf{H} = \begin{bmatrix} h_1^{(1)} & h_2^{(1)} \\ \vdots & \vdots \\ h_1^{(n_c)} & h_2^{(n_c)} \end{bmatrix}, \mathbf{l} = \begin{bmatrix} l^{(1)} \\ \vdots \\ l^{(n_c)} \end{bmatrix}$$

Terminal velocity constraint

In our work, the additional velocity constraint

$$0 \le \sqrt{v_{\mathrm{P}x}^2(k+N-1) + v_{\mathrm{P}y}^2(k+N-1)} \le \Delta v_{\max} \tag{14}$$

is enforced on the control variables associated to the last step of the prediction horizon. This constraint plays the role of a terminal constraint. In fact, importantly, it ensures that the predicted velocity of the wheelchair at time $k+N-1$ (where the prediction takes place at time k) is physically compatible with a possible sudden stop at instant $k+N$ (i.e., at the end of the control horizon) when approaching the obstacles. In this way, when fixed obstacles enter the prediction range, as the corresponding constraints becomes active, a feasible solution is therefore guaranteed to be available. This guarantees recursive feasibility of the MPC-related optimization problem.

Note that, if N is sufficiently large, constraint (14) does not limit the maximum speed reachable at each time instant by the wheelchair, i.e., constraint (14) does not prevent v_{long} to attain $v_{\max}$.

For inclusion in the optimization problem, the following linearized inequalities are used, derived similarly to (7).

$$\begin{aligned} 0 \le &\frac{v_{\mathrm{P}x}(k+N-1|k-1)}{\bar{v}(k+N-1|k-1)} v_x(k+N-1) \\ &+ \frac{v_{\mathrm{P}y}(k+N-1|k-1)}{\bar{v}(k+N-1|k-1)} v_{\mathrm{P}x}(k+N-1) \le \Delta v_{\max} \end{aligned} \tag{15}$$

Finally, note that $v_{\mathrm{P}x}(k+N-1|k-1)$ and $v_{\mathrm{P}y}(k+N-1|k-1)$ are not given as part of the solution profile of the MPC optimization at time $k-1$. Indeed, they are defined as $v_{\mathrm{P}x}(k+N-1|k-1) = v_{\mathrm{P}x}(k+N-2|k-1)$ and $v_{\mathrm{P}y}(k+N-1|k-1) = v_{\mathrm{P}y}(k+N-2|k-1)$. Similarly to the other constraints, the latter are included in the optimization problem as follows

$$\begin{aligned} &\begin{bmatrix} \frac{v_{\mathrm{P}x}(k+N-1|k-1)}{\bar{v}(k+N-1|k-1)} & \frac{v_{\mathrm{P}y}(k+N-1|k-1)}{\bar{v}(k+N-1|k-1)} \\ -\frac{v_{\mathrm{P}x}(k+N-1|k-1)}{\bar{v}(k+N-1|k-1)} & -\frac{v_{\mathrm{P}y}(k+N-1|k-1)}{\bar{v}(k+N-1|k-1)} \end{bmatrix} \mathbf{u}(k+N-1) \\ &\le \begin{bmatrix} \Delta v_{\max} \\ 0 \end{bmatrix} \end{aligned} \tag{16}$$

3.3. Slack variables

A remark is due at this point. The real vehicle state evolution may be subject to a slight mismatch with respect to the nominal equation

(4) derived in Section 2; the reason of this mismatch can be manyfold, e.g., discretization and linearization errors, possible numerical inaccuracies in the implementation of the feedback linearization scheme, and actuator model identification and approximation errors. Also, other sources of nonideality may be present, e.g., inaccurate position estimate by the odometry algorithm and possible inaccuracies in the estimation of the distance and of the orientation of the obstacles. For this reason it is common practice to take a twofold action on the position limitation (13): (i) constraint tightening and (ii) constraint softening through the use of slack variables. We resort to slack variables also in case of the limitations in the velocity variation: indeed, we may need to violate comfort requirements in case of danger, e.g., in case an object is moving towards the wheelchair.

First, the obstacle avoidance constraint (13) is now rewritten, for all $i = 1, \dots, N$, as

$$\mathbf{H}\xi(k+i) \leq \mathbf{l} - \Delta l_{\text{diag}}(\mathbf{1} - \mathbf{u}_{\text{slack},\xi}) \tag{17}$$

where Δl_{diag} =diag$(\Delta l_1, \dots, \Delta l_{n_c})$, and $\Delta l_j = \Delta l^o \sqrt{(h_1^{(j)})^2 + (h_2^{(j)})^2}$; Δl^o is the absolute "security distance" from the obstacle boundary that, if possible, should respected. Also, $\mathbf{1}$ is a unity vector with n_c components, and $\mathbf{u}_{\text{slack},\xi}$ is a n_c-dimensional vector of slack variables, each included in the range $[0, 1]$: the latter condition makes it necessary to introduce the further set of constraints

$$\begin{bmatrix} I_{n_c} \\ -I_{n_c} \end{bmatrix} \mathbf{u}_{\text{slack},\xi} \leq \begin{bmatrix} \mathbf{1} \\ 0_{n_c \times 1} \end{bmatrix} \tag{18}$$

where $0_{n_c\times 1}$ is a zero vector with n_c components. Secondly, we introduce a slack vector $\mathbf{u}_{\text{slack},\Delta v}$ with two positive entries for allowing violations to constraints (11), and the resulting constraint set consists of

$$\begin{bmatrix} -1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} \mathbf{u}(k+i-1) \\ \mathbf{u}(k+i) \end{bmatrix} \leq \Delta v_{\max} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \mathbf{u}_{\text{slack},\Delta v} \tag{19}$$

where the additional constraint is

$$-\mathbf{u}_{\text{slack},\Delta v} \leq 0_{2\times 1} \tag{20}$$

The slack variables will be minimized if possible in the cost function, as described in the following section.

3.4. Formulation of the MPC optimization problem

We are now in the position to formulate the MPC optimization problem to be solved at the discrete-time instant k:

$$\min_{\mathbf{u}(k),\dots,\mathbf{u}(k+N-1),\mathbf{u}_{\text{slack},\xi},\mathbf{u}_{\text{slack},\Delta v}} J(k) + \|\mathbf{u}_{\text{slack},\xi}\|^2_{Q_\xi} + \|\mathbf{u}_{\text{slack},\Delta v}\|^2_{Q_{\Delta v}}$$

where $J(k)$ is given in (5) and Q_ξ, $Q_{\Delta v}$ are suitable weights on the slack variables; in general, the entries of $Q_\xi, Q_{\Delta v}$ are significantly greater than the ones of $\mathbf{Q}$ and $\mathbf{R}$ to allow for the use of slack variables only when strictly necessary.

The problem is subject to the following constraints.

- System dynamics (4), where the initial condition $\xi(k)$ is the measured/estimated state of point P at the current discrete-time instant k.
- Position constraints (17) and (18).
- Absolute speed constraints: (8) (or (9), for velocities which are close to zero) for all $i = 0, \dots, N-2$ and (16) for $i = N-1$.
- Speed variation constraints (19) and (20) for all $i = 0, \dots, N-1$.

The result of the optimization problem, at time step k, consists of the optimal values of $\mathbf{u}(k)$, …, $\mathbf{u}(k+N-1)$, $\mathbf{u}_{\text{slack},\xi}$, $\mathbf{u}_{\text{slack},\Delta v}$: only the first component $\mathbf{u}(k)$ of the optimal input sequence is applied to the (feedback-linearized) system depicted in Fig. 4. Then according to the receding horizon principle (Mayne et al., 2000), at the next time instant $k+1$ new data are collected and the procedure sketched in this section is repeated again.

4. Control architecture and implementation issues

This section is devoted to implementation details, including the implementation of the control system using a ROS-based architecture, the algorithm used for control of the wheelchair orientation, the procedure employed to define the constraint set from laser scanner data, and the algorithm used for obstacle circumnavigation based on vector-field reference setpoints.

4.1. ROS navigation stack

The proposed control system has been implemented using the ROS publisher–subscriber paradigm (Quigley et al., 2009). Dedicated nodes, running in parallel, perform computations and communicate with each other through topics, i.e., buses dedicated to message exchange.

ROS provides the navigation stack, a standard software architecture designed to be as general-purpose as possible: it takes in information from odometry, sensor streams, and velocity commands and allows to perform the following tasks.

- *Mapping*, required to set up a map of an initially unknown environment. This problem is addressed mainly by the map server[1] and gmapping[2] packages.
- *Localization*, to estimate where the robot is located in a map. The problem is addressed by the amcl package,[3] fed by a map and laser scan measurements.
- *Planning*, to determine paths or trajectories that allow a robot to move in an environment avoiding obstacles. The move base package[4] handles the planning phase providing global and local planning algorithms. Planning algorithms allow to compute a path or a trajectory connecting a start and a goal pose (global planner) and to control the robot during path or trajectory tracking (local planner).

A graphical representation of a typical navigation stack setup is shown in Fig. 6.

The global and local planners, used by the *move_base* node, can be any algorithms compliant with the interface specified in the nav_core package.[5] The move_base node also provides two cost maps,[6] one for the global planner and one for the local planner, that store all the environmental information required to perform a navigation task. Every time a new goal is selected, the global planner computes a plan connecting the current robot position to its goal using the global map. On the other hand, the local planner only deals with the immediate surroundings of the robot taking into consideration both the plan and the data provided by sensors. Thanks to it, the robot is driven on a trajectory that best approximates the path provided by the global planner while, at the same time, ensuring no collisions.

The navigation stack implementation used to perform the tests presented in this paper is based on the standard global planner, i.e., Dijkstra algorithm. The global plan is then decomposed into intermediate goals, used as reference positions ξ_{ref} by the MPC controller, which acts as local planner. The sequencing of the intermediate goals is handled in an appropriate way in order to avoid discontinuity in the velocity. Eventually, as described in the following Section 4.2, the final orientation control is executed only when the final goal position is achieved.

[1] http://wiki.ros.org/map_server.
[2] http://wiki.ros.org/gmapping.
[3] http://wiki.ros.org/amcl.
[4] http://wiki.ros.org/move_base.
[5] http://wiki.ros.org/nav_core.
[6] http://wiki.ros.org/costmap_2d.

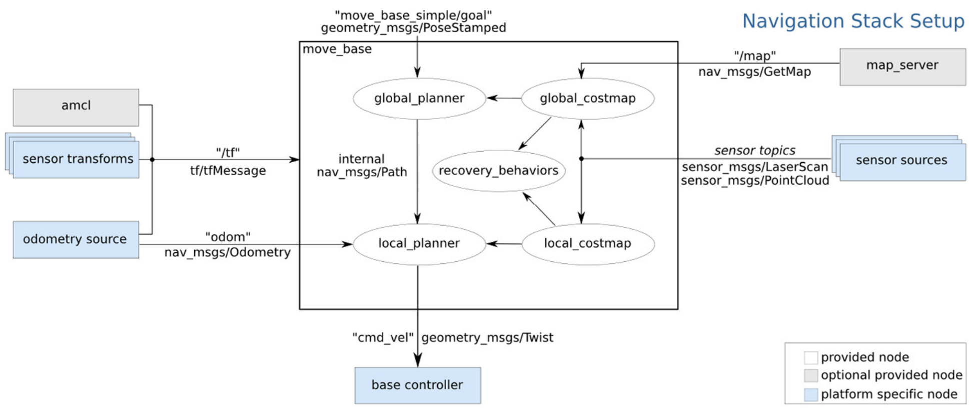


Fig. 6. ROS navigation stack.

4.2. Orientation control phase

The control scheme presented in the previous Section 3 has the objective to steer the wheelchair to a predefined goal position. The wheelchair orientation is therefore defined, at each time instant, by the velocity direction (i.e., by the references v_{Px} and v_{Py}), as also analyzed in Appendix. It may be necessary, in many real cases, to assume a desired orientation at rest position. To do it, a different (linear and proportional) control algorithm takes over when the final goal is achieved. More specifically, the orientation control law sets $v = 0$, while the value taken by the input ω is proportional to the angle displacement with respect to the desired one.

4.3. Sensor processing and definition of the feasible set

In this section we describe the algorithm used to compute, in real-time, constraints (12) from laser scanner data. The available measurements are, besides the wheelchair pose in the global frame, the laser scans from the built-in sensors. More specifically, the measurements are points scanned, with a 360 degrees view range and a resolution of 0.33 degrees (see Section 5.1), i.e., a point cloud in the wheelchair reference frame. The processing algorithm consists of the following four steps.

1. A roto-translation from the wheelchair to the global reference frame, i.e., the same used for the control problem definition, is applied to the point cloud;
2. Points are grouped into clusters according to their relative position in order to distinguish obstacles from spot openings in the environment. Namely, if the Euclidean distance between two consecutive points is larger than a predefined threshold (i.e., which is reasonably larger than the width of the wheelchair) the two points are regarded as belonging to different clusters (i.e., different objects). Thanks to this, the vehicle is allowed to pass in the space between the two perceived obstacles.
3. Clusters are simplified by removing outliers and all redundant points for the obstacle outline. Note that each object's surface has to be convex. For this reason, clusters including concavities are preliminarily split into a number of convex subsets. Fig. 7 shows the resulting segments highlighted in red.
4. A linear inequality constraint is selected for each object.

The constraint selection has the aim of defining the convex region, as in Fig. 7, that somehow maximizes the working space area without violating the detected obstacle footprints. Among the lines characterizing the boundary of each obstacle, at each time step we take the one that is closest to the current position of the vehicle, as this is the line that divides the plane into two sub-planes, one including the obstacle and one including the wheelchair. If this line, however, reduces the state space area too much, a situation that occurs when the wheelchair is facing a

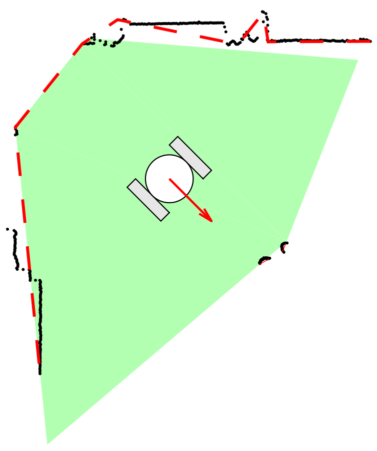


Fig. 7. Obstacle detection process. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

corner rather than an edge of the obstacle, the chosen constraint is a line passing through the corner and orthogonal to the distance between the line itself and the current position of the wheelchair.

4.4. Vortex field

While the constraints defined in Section 4.3 allow for collision avoidance, they do not guarantee the vehicle to reach the goal position in general. In case a concave cluster of objects (especially if not included in the environment map embedded in the wheelchair software) is placed between the current wheelchair position and the goal, a deadlock situation may occur. In this case the vehicle may get stuck in the proximity of the obstacle, i.e., in the position that allows to minimize the cost function on the considered control horizon.

To overcome this issue, different approaches are possible. First, enlarging the control horizon may definitely help to avoid many deadlock situations but, on the other hand, this would increase the computational effort. As a possible solution, whenever an obstacle is close to the wheelchair (i.e., if the current distance $d(k)$ is smaller than a predefined value d_{VF}) and in case it lies between the wheelchair and the goal position, a different reference may created (online, i.e., at each time instant) allowing to navigate around the obstacle (see Fig. 8). Such alternative reference is defined, at each time step, as

$$\xi_{ref}(k) = \xi(k) + v_{VF}(k)$$

where the vector $v_{VF}(k)$ depends on the definition of a corresponding repulsive cost, according to the *Vortex Field* theory (De Medio et al.,

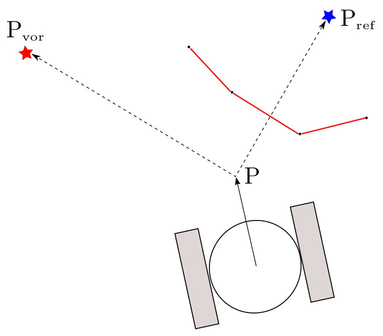


Fig. 8. *Vortex field* triggering logic.

1991), similarly to Buizza Avanzini et al. (2015). In our case, $v_{\mathrm{VF}}(k)$ is parallel to the selected obstacle side to be overcame, with a direction defined according to the vehicle current orientation and with absolute value that is inversely proportional to the current distance from the obstacle, i.e., $|v_{\mathrm{VF}}(k)| = k_{\mathrm{VF}} \frac{d_{\mathrm{VF}}}{d(k)}$. Note that, since $d(k) < d_{\mathrm{VF}}$, the constant k_{VF} is the minimum length of v_{VF}.

5. Experimental results

This section presents the results of a set of experiments performed using a commercial wheelchair, equipped with suitable sensors, in realistic scenarios.

In particular, the effectiveness of the proposed MPC local planner and its integration in the ROS navigation architecture introduced in Section 4.1 are discussed.

5.1. Experimental setup and testing scenario

The experimental setup is composed of:

- an electric wheelchair *Degonda Twist t4* 2×2 (see Fig. 9), produced by *Degonda Rehab SA*, having two independent driving wheels actuated by 350 W electric motors, and three stabilizing caster wheels. The wheelchair is equipped with a CAN-bus, allowing peripherals to communicate to each other.
- a slim computer, DS81L with Haswell Refresh Core i7 and 8 Gb memory, for automotive and robotic applications, that serves as on board computing system.
- a Chipset Module, a dedicated board produced by PGDT, allowing the computer to be interfaced to the communication bus.
- two rotary encoders providing information about the angular velocity of the two wheels.
- two time-of-flight laser scanners Sick TiM 561 (see Fig. 10), detecting the position of the objects in the surrounding environment. Each scanner has an angular aperture of 270 deg, a standard range of 8 m, and a resolution of 0.33 deg. The two scanners are mounted at the front left and rear right corner of the wheelchair so as to ensure that for a radius of 8 m around the robot no blind areas exist (see Fig. 10).

The tests performed to validate the MPC control strategy have been carried out in a realistic scenario represented by the ground floor of a university building (see Fig. 11). This scenario is mainly composed of a corridor approximately 55 m long with a square room in the middle and some doors on the left and right side. The grayed area in Fig. 11 shows the part of the map where the trajectories reported in the following have been performed.

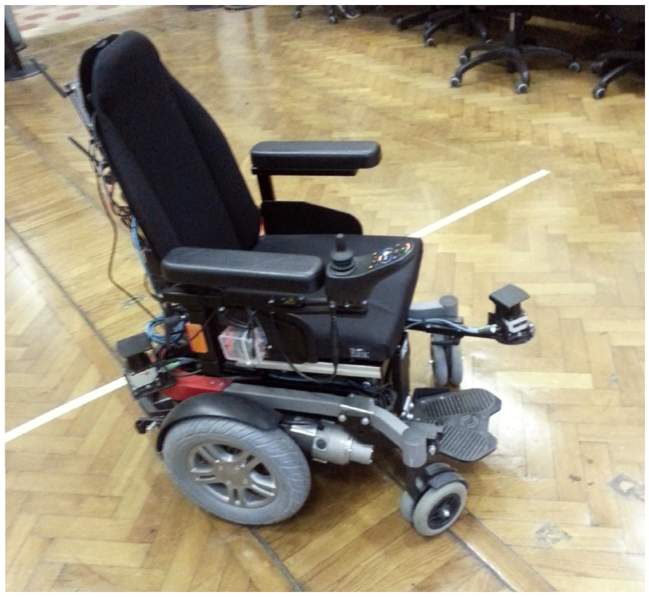


Fig. 9. The electric wheelchair Degonda Twist t4 2×2.

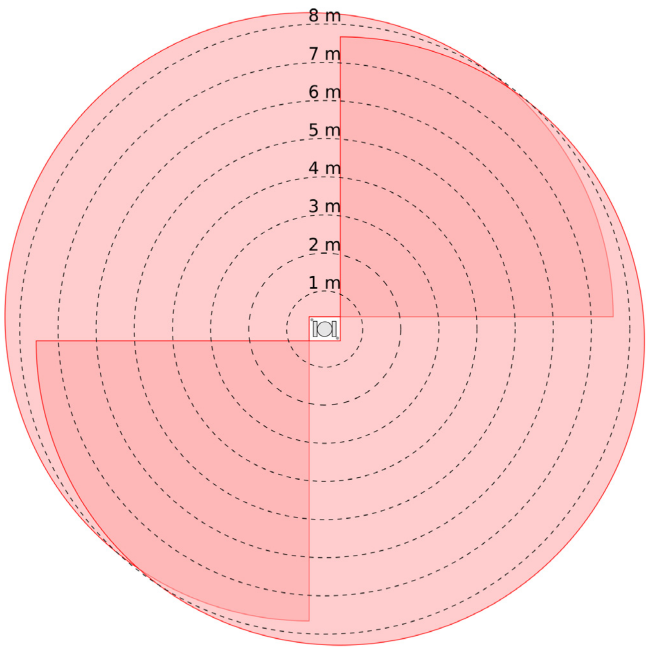


Fig. 10. Laser scanner field of view (dashed circles centered in the center of the wheelchair represent the visibility range at different distances).

5.2. Controller parameter selection

This section describes how the controller parameters have been selected. Note that parameter tuning has been conducted in a preliminary phase, before the implementation in the ROS navigation stack. In particular, the selected parameter values are the ones discussed in Ceravolo et al. (2017).

The first tuning knob corresponds to the length ε, used in the feedback linearization procedure. Experimental tests show that, the larger ε, the smoother are the wheelchair dynamics but, on the other hand, a large value of ε would make it difficult to define, in a conservative way, the obstacle avoidance constraints. A good compromise has been found

Table 1
Controller parameters.

Parameter	Value	Description
μ_x	0.0225	Actuator gain ω/x_J
μ_y	0.007871	Actuator gain v_{long}/y_J if $y_J \geq 0$
μ_y	0.009444	Actuator gain v_{long}/y_J if $y_J < 0$
ε	0.5 m	Distance of point P from the wheel axle center
τ_s	0.2 s	Sampling time
$v_{\max}$	0.55 m/s	Maximum longitudinal velocity
v_{low}	0.05 m/s	Threshold defining the low-speed region
$\Delta v_{\max}$	0.04 m/s	Maximum velocity variation (on x and y axes)
$\mathbf{Q}$	I_2	State weight in cost (5) when ξ_{ref} is a goal position
$\mathbf{Q}$	$10I_2$	State weight in cost (5) when ξ_{ref} is vector field-generated
$\mathbf{R}$	$5I_2$	Input weight matrix in cost (5) (all cases)
N	15	Prediction/Control horizon
Δl^o	0.2 m	Security distance for the obstacle avoidance slack variable
Q_ξ	10^9	Weight on the slack variable $\mathbf{u}_{\text{slack},\xi}$ in the cost function
$Q_{\Delta v}$	10^3	Weight on the slack variable $\mathbf{u}_{\text{slack},\Delta v}$ in the cost function
d_{VF}	2.5 m	Distance of activation of the vector field
k_{VF}	5 m	Vector field-related constant

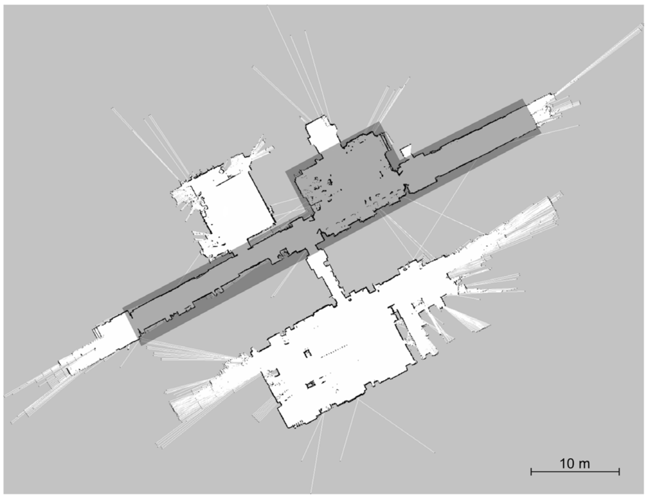


Fig. 11. A map of the environment where tests have been performed.

in $\varepsilon = 0.5$ m, making the point P actually correspond to the wheelchair footboard.

Two other major parameters to be properly defined are the sampling time τ_s and the prediction horizon length N, which are strictly related to each other. Importantly, the product $T_{\text{ph}} = \tau_s N$, i.e., the actual prediction time span, has to be greater than the time T_{halt} required to stop the wheelchair when the vehicle is traveling at the maximum speed, i.e.,

$$\tau_s N = T_{\text{ph}} > T_{\text{halt}} = \frac{v_{\max}}{a_{\max}} \tag{21}$$

In our case $v_{\max} = 0.55$ m/s and $a_{\max} = 0.2$ m/s^2. Parameters N and τ_s must therefore be tuned under constraint (21), and in order to find a good compromise between feasibility of the optimization problem and performance. Indeed, decreasing τ_s the controller becomes more and more reactive; however, the prediction horizon N must correspondingly be larger and larger: this would in turn increase the size of the optimization problem and its computational complexity. On the other hand, a larger sampling time would make the optimization problem more lightweight, but at the price of a less responsive controller. A good compromise is $N = 15$, $\tau_s = 200$ ms. In this case, the average (respectively maximum) time required to solve the MPC optimization problem, at each time step, is 11.3 ms (respectively, 34 ms). Just for comparison, if $N = 30$ and $\tau_s = 100$ ms, the average (respectively, maximum) time required for computing the solution to the MPC problem is 46.4 ms (respectively, 128 ms), resulting impractical.[7] A comprehensive analysis, including also the impact of the constraints on the computational time, has been conducted in Ceravolo and Gabellone (2015–2016).

Regarding the parameters used to define the constraints in Section 3.2, we have set $v_{\text{low}} = 0.05$ m/s and $\Delta v_{\max} = a_{\max}\tau_s$. Also, regarding the parameters used in Section 3.3, $\Delta l^o = 0.2$ m.

The cost function weights $\mathbf{R}$ and $\mathbf{Q}$ mainly affect the transient behavior of the system, i.e., where constraints are not active. A higher penalty on the position error leads to faster speed transient and, also due to possible delays in the actuation system, it can result in oscillations. In line with this consideration, we set $\mathbf{R} = rI_2$ and $\mathbf{Q} = qI_2$, where $r = 5$ and $q = 1$, where I_2 is the 2×2 identity matrix. This choice corresponds to smooth but relatively fast transients. The matrix $\mathbf{P}$ is derived by $\mathbf{Q}$ and $\mathbf{R}$ as the solution to the Lyapunov equation $(\mathbf{A}+\mathbf{B}\mathbf{K}_{\text{aux}})^T\mathbf{P}(\mathbf{A}+\mathbf{B}\mathbf{K}_{\text{aux}})-\mathbf{P} = -(\mathbf{Q}+\mathbf{K}_{\text{aux}}^T\mathbf{R}\mathbf{K}_{\text{aux}})$, where $\mathbf{K}_{\text{aux}}$ is a suitable stabilizing auxiliary controller gain. Setting $\mathbf{K}_{\text{aux}} = k_{\text{aux}}I_2$, where $k_{\text{aux}} = -1/(2\tau_s)$, we have that $\mathbf{P} = pI_2$, where $p = 4/3(q + r/(4\tau_s^2))$.

The weights corresponding to the slack variables are $Q_\xi = 10^9 I_{n_c}$ and $Q_{\Delta v} = 10^3 I_2$.

As for the vortex field, the tuning knobs are the distance threshold d_{VF} and the minimum vector length k_{VF}. The former parameter is selected in order to be greater than the distance required to stop the robot when it moves at the maximum speed, i.e.

$$d_{\text{VF}} = 2.5 \text{ m} > d_{\text{stop}} = \frac{v_{max}^2}{2a_{max}} = 0.75625 \text{ m}$$

The value of k_{VF} must also be greater than d_{stop} to avoid decelerations when the vortex field reference is activated, i.e., $k_{\text{VF}} = 5$ m. In this case we also increase the weight on the state vector, i.e., we set $q = 10$, while $r = 5$ as before.

For better clarity, a summary of the control parameters is given in Table 1.

5.3. Navigation experiments

To test the behavior of the proposed controller a number of real experiments has been carried out in the environment described in Section 5.1. This section describes the results of the most relevant of those experiments, some considering only static obstacles, others including moving obstacles as well.

Fig. 12 shows the results of four different navigation experiments, the first three of them considering only static obstacles. The blue lines represent the trajectories generated by the global planner from the

[7] IBM ILOG CPLEX Optimization Studio is used for solving the optimization problem.

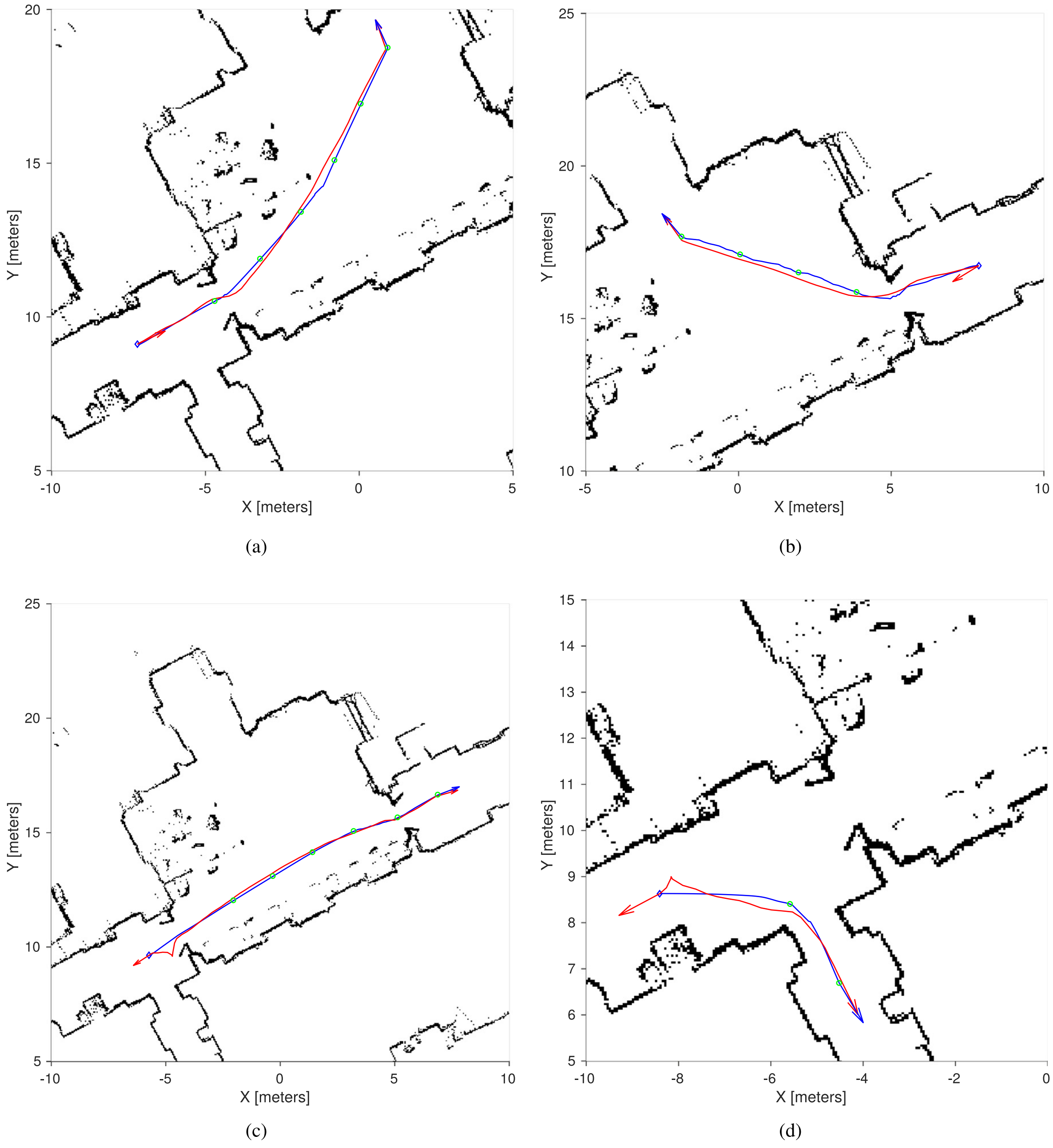


Fig. 12. Four different navigation experiments. Global plan (blue line) and wheelchair trajectory (red line), the blue diamond represents the starting position, the green circles the intermediate goals. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

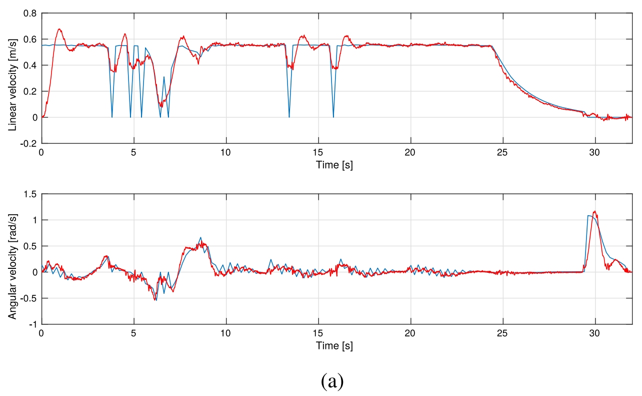
initial wheelchair positions (the blue diamonds in Fig. 12) to the selected final poses, whose orientations are represented by the blue arrows. The green circles, instead, are the intermediate goals generated by the local planner and used by the MPC controller as reference positions. Finally, the red trajectories represent the paths traveled by the wheelchair, and the red arrows the final wheelchair orientations.

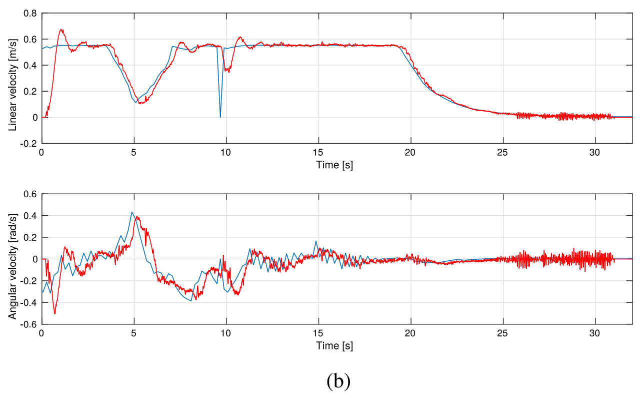
As can be seen from Fig. 12 the wheelchair trajectory is almost always very close to the global path and passes by the intermediate goals. It must be noticed, however, that the global path is generated only minimizing the distance between the start and goal positions, without considering the kinodynamic constraints characterizing the wheelchair, the safety distance from obstacles enforced by the MPC controller, and the fact that the controller does not allow to directly control the wheelchair orientation. For this reason, the planned and executed trajectories can be in some specific situations, like for example in the case of high curvature paths, obstacle premises, etc., not so close.

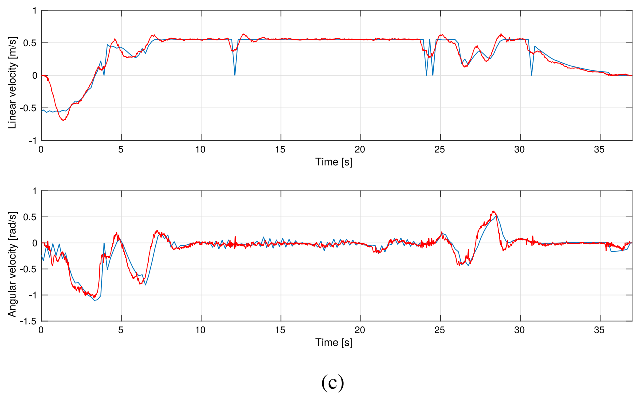
In particular, considering the experiments shown in Fig. 12, it must be noticed that[8] :

- in the first experiment (Fig. 12(a)), the MPC controller follows a shorter and smoother path, with respect to the one generated by the global planner, as the global plan is influenced by the presence of small obstacles in the map that are no more present in the actual environment;
- in the second experiment (Fig. 12(b)), the MPC controller follows a shorter and smoother path with respect to the global plan that is more discontinuous due to the map gridding operated by Dijkstra algorithm;

[8] For the interested reader, a video showing the behavior of the wheelchair, as can be seen from rviz, during the four experiments here described can be found at https://youtu.be/MiVn0KmqowM Another video showing the wheelchair during different maneuvers can be found at https://youtu.be/b9cqhRTQgA0.







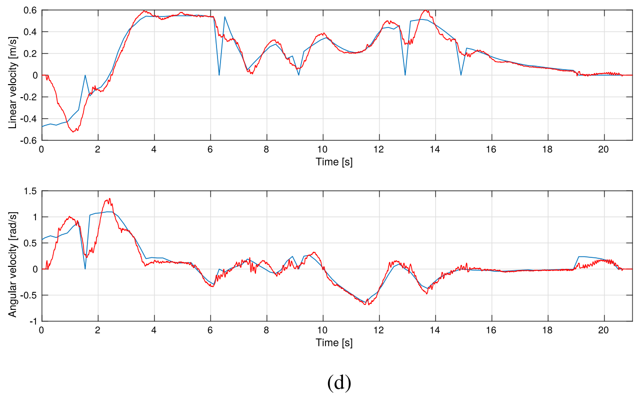


Fig. 13. Wheelchair longitudinal and angular velocity during the four experiments in Fig. 12 (blue lines represent reference velocities, i.e., inputs to the "Actuator^{-1}" block in Fig. 4, red lines represent actual velocities). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

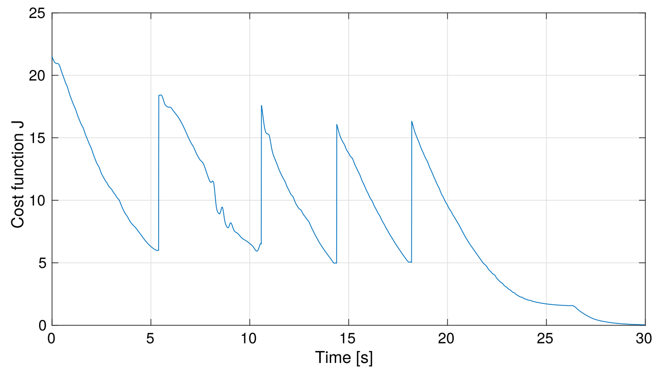


Fig. 14. Time history of the cost function in the first experiment.

- in the third experiment (Fig. 12(c)), the actual path is initially different from the global plan as the wheelchair, that starts with the wrong orientation, needs to perform a maneuver to be oriented toward the goal and tangent to the plan;
- in the fourth experiment (Fig. 12(d)), the actual path is initially different from the global plan due to the presence of moving obstacles in the scene that are not considered by the map.

In Fig. 13 the wheelchair velocities are depicted. The blue lines represent the linear and angular velocity commands determined by the MPC, the red lines the actual linear and angular velocities generated by the wheelchair motors.

The time histories of the velocities show the responsive but rather smooth approach of the robot towards the goal, and the fact that both velocities satisfy the actuator constraints. Note that, in all the

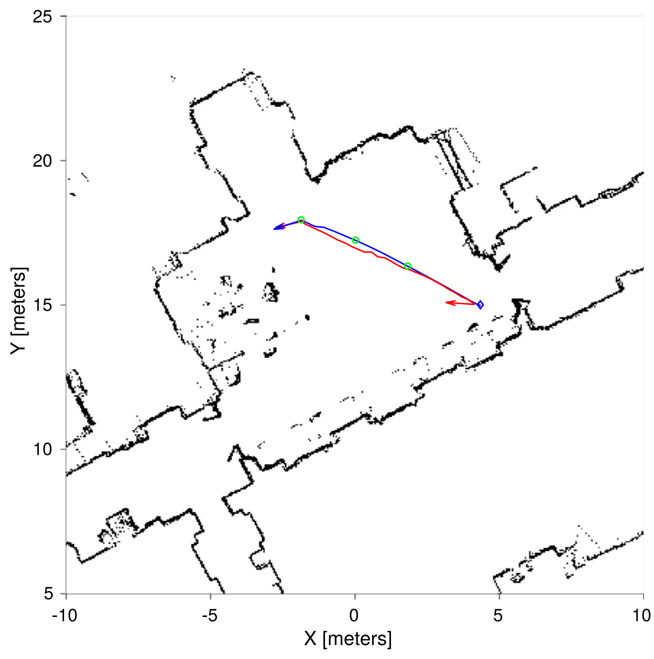


Fig. 15. A navigation experiment without obstacle detection. Global plan (blue line) and wheelchair trajectory (red line), the blue diamond represents the starting position, the green circles the intermediate goals. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

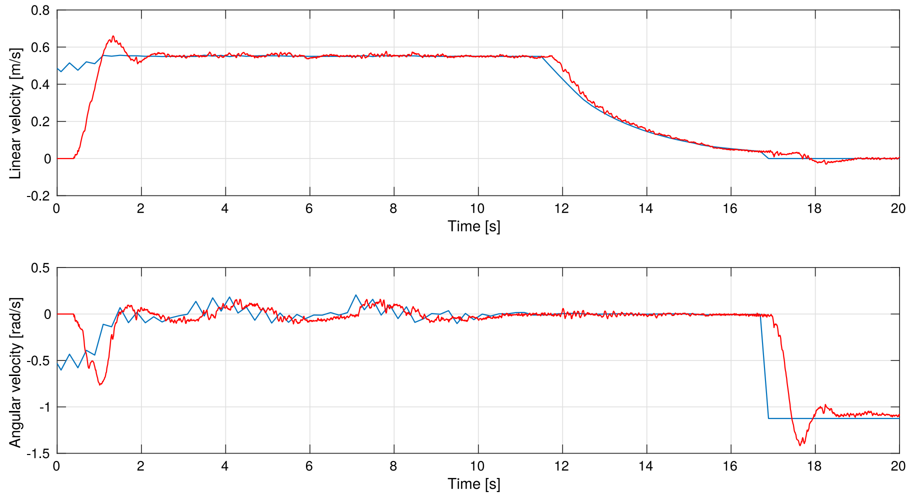


Fig. 16. Wheelchair longitudinal and angular velocity in the experiment in Fig. 15 (blue lines represent reference velocities, i.e., inputs to the "Actuator^{-1}" block in Fig. 4, red lines represent actual velocities). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

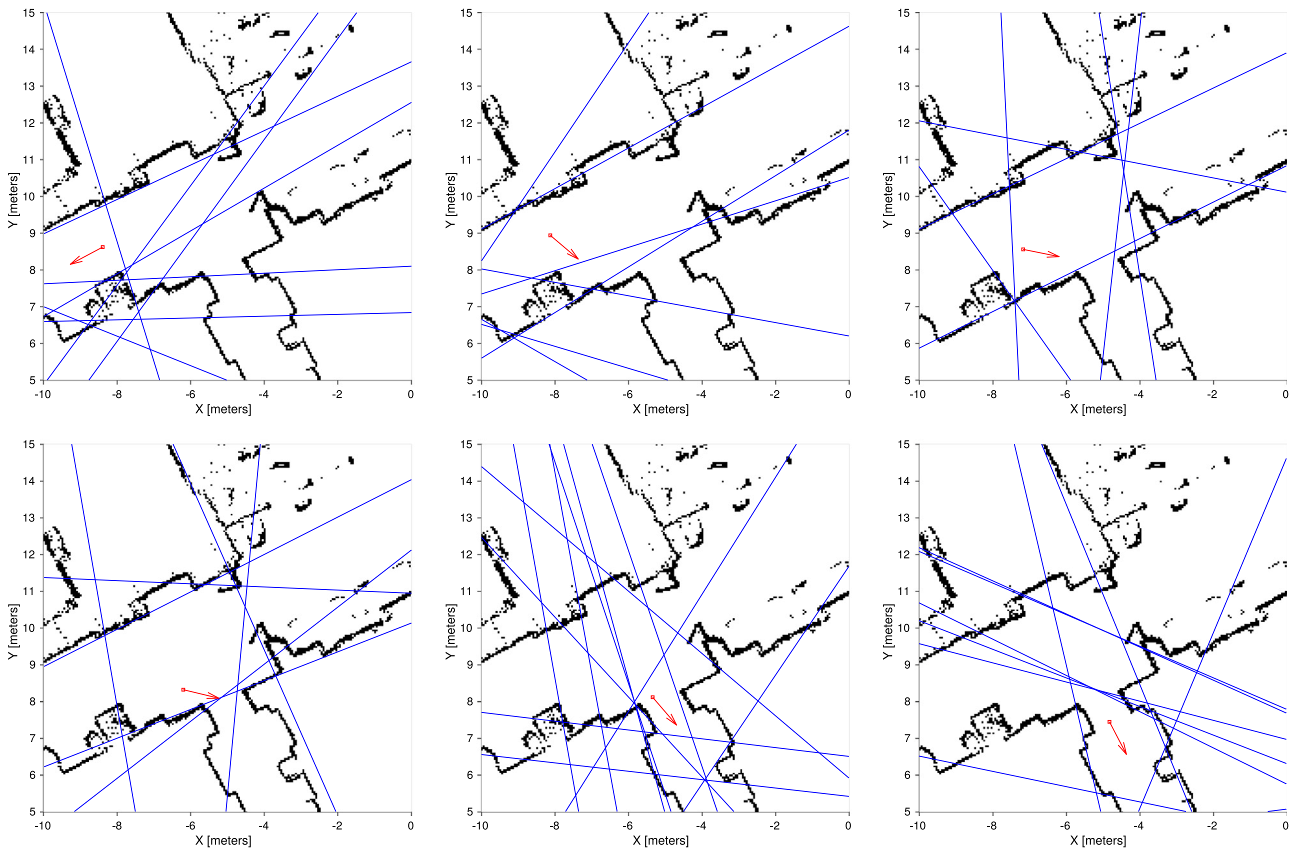


Fig. 17. Obstacle avoidance constraints generated during the execution of the global plan in Fig. 12(d). The red square and arrow represents the wheelchair pose. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

experiments the linear velocity is characterized by some isolated time instants where it suddenly goes to zero. This effect is due to noise in the laser scanner measurements generating erroneous obstacle constraints.

As a counter-check, an experiment has been executed navigating in the free space without obstacle detection (Fig. 15). As can be seen from Fig. 16 no isolated zero velocity points are present in this case.

Fig. 14 shows, for the first experiment, the time history of the cost function $J(k)$. Note that, a discontinuity occurs every time a new intermediate goal is selected.

Finally, in all the experiments the linear velocity command computed by the MPC is different from zero since the first time instant. This is due to the fact that the MPC computation starts immediately after a new goal has been selected and a global plan has been computed, while, for safety reasons, the wheelchair motors start only after being manually enabled.

Focusing now on the fourth experiment (Fig. 12(d)), it must be noticed that the wheelchair is able to correctly maneuver even in a very small space and enters a door less than 1.2 m wide. Due to the importance

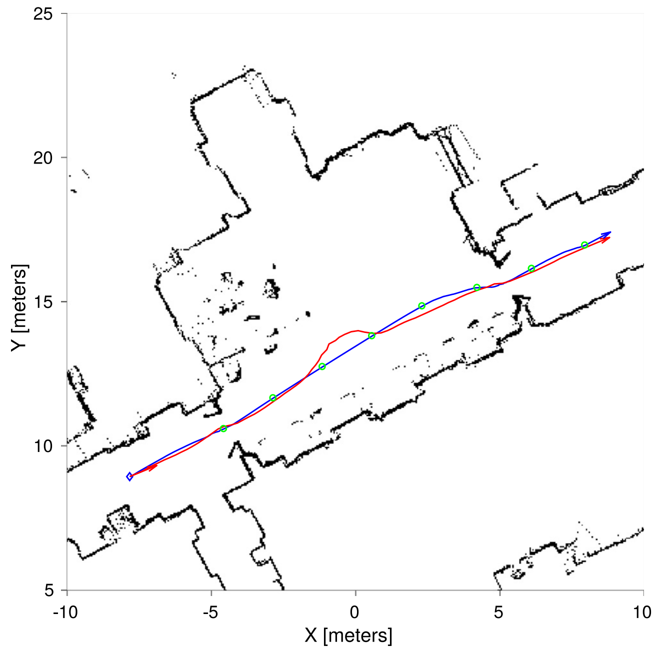


Fig. 18. A navigation experiment with moving obstacles. Global plan (blue line) and wheelchair trajectory (red line), the blue diamond represents the starting position, the green circles the intermediate goals. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of the obstacle constraints in executing this maneuver, we have reported a set of screen-shots (Fig. 17) showing the wheelchair pose (red square and arrow) and the evolution of the constraints (blue lines).

A further experiment has been performed in order to show the behavior of the MPC controller in the presence of two persons (green crosses in Fig. 20, corresponding to persons' legs as seen by the laser scanners) moving towards the wheelchair.

The wheelchair, starting from the left corridor, has to follow an almost straight trajectory crossing the room and moving towards the right corridor (Fig. 18). As soon as it enters the room, two persons coming from the right side walk against it (Fig. 20). The wheelchair turns to the left, leaving the planned path, but it immediately faces another obstacle on its left side, a person slowly walking to the right. The wheelchair turns right again to avoid the person and comes back to the planned path.

From Fig. 19 it is evident that the wheelchair reduces the linear velocity three times, i.e., at time 16 s, 21 s, and 31 s, to execute a turning maneuver in order to avoid the two persons coming from its right side, to come back to the planned path, and to avoid the person coming from its left side, respectively.

Finally, Fig. 20 shows the evolution of the obstacle constraints during the time instants immediately before and after the appearance of the moving obstacles.

The last experiment demonstrates that, though only static obstacles have been considered in the design of the controller, the MPC is able to guarantee obstacle avoidance even in the case of slowly (compared to the sampling time) moving ones.

As the controller lacks a motion model of the humans walking in the scene, it is not able to guarantee obstacle avoidance anymore in case, for example, of a running person. On the other side, introducing a motion model allows the MPC to predict the human trajectory in the prediction time span, increasing the control system performance. This will constitute a future research direction.

6. Discussion and conclusions

In this paper, a linear MPC control scheme has been proposed to address the motion problems of an autonomous wheelchair in a realistic Ambient-Assisted Living (AAL) environment.

Though the kinematic model of the wheelchair is represented by nonlinear dynamics, the introduction of an inner loop based on a feedback linearizing law simplifies the development of the controller and allows for a real-time computationally-efficient implementation. On the other side, thanks to the MPC framework, operational constraints, like obstacle avoidance, actuator limitations and passenger comfort, can be easily included in the optimization problem. Experimental results show the effectiveness of the proposal, and demonstrate that MPC can be considered a promising control approach to develop autonomous vehicles.

This work will be extended in several directions. First, tests will be conducted on the stress levels of the users, in order to define, in a more rigorous and quantitative way, the correlation between acceleration and velocity bounds and the user tension. Based on this, we can define different modes of operation, related to different user categories and needs. Another future extension is related to the possibility of outdoor operation: this will require to define and test the most appropriate outdoor localization algorithms for our case, including dGPS and inertial measurement unit's information. Also, this will require to define, in a

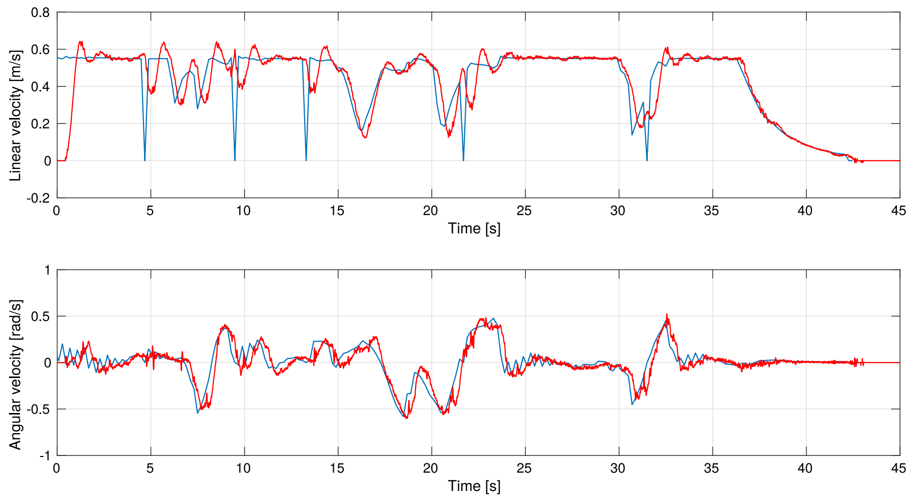


Fig. 19. Wheelchair longitudinal and angular velocity in Fig. 18 (blue lines represent reference velocities, i.e., inputs to the "Actuator^{-1}" block in Fig. 4, red lines represent actual velocities). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

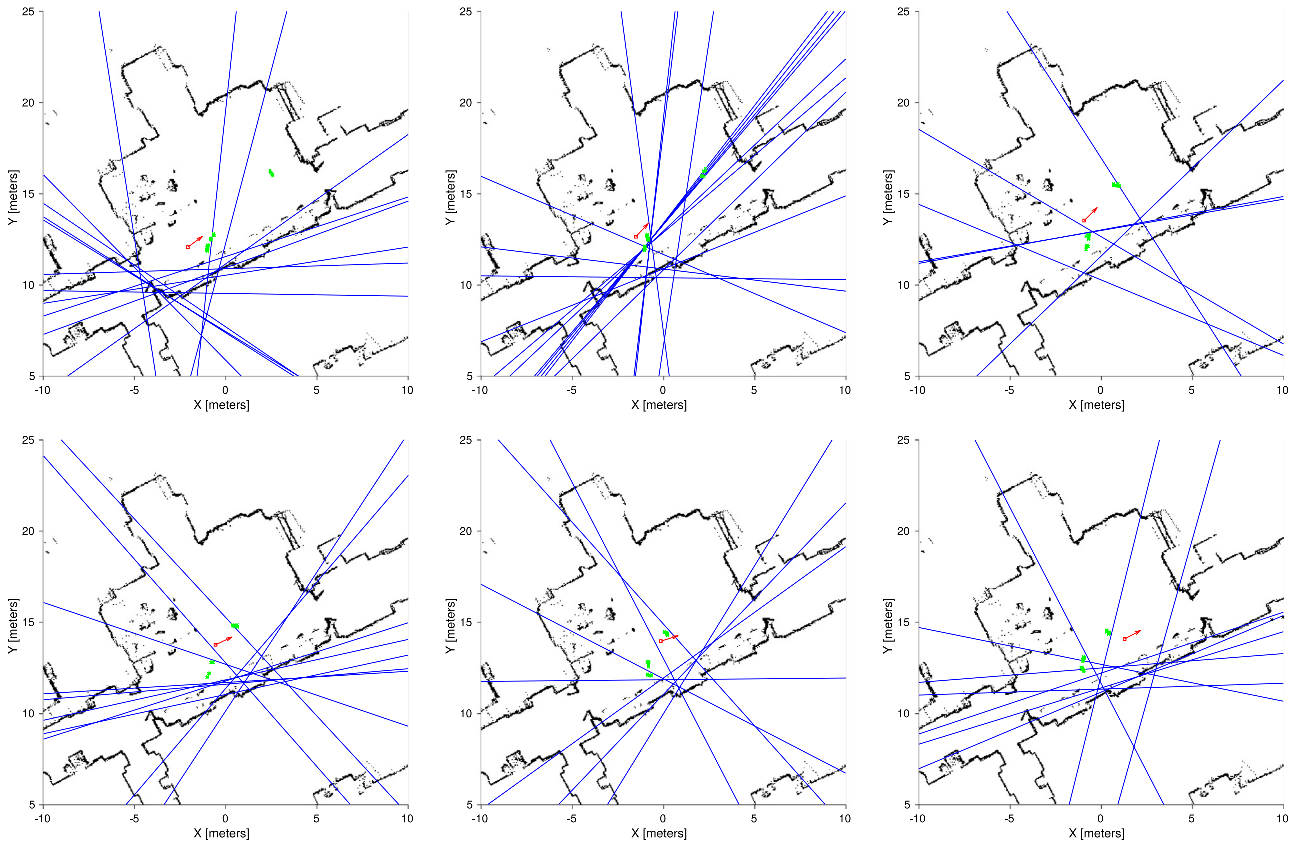


Fig. 20. Obstacle avoidance constraints generated during the execution of the global plan in Fig. 18. The red square and arrow represents the wheelchair pose. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

rigorous way, strategies for collision avoidance with respect to moving objects and pedestrians. This will be done, for example, incorporating pedestrian stochastic models, similarly to De Nicolao, Ferrara, and Giacomini (2007), and applying a stochastic MPC scheme (see Farina, Giulioni, & Scattolini, 2016 for a recent survey) to fulfill probabilistic-type bounds.

Acknowledgment

This work has been partially supported by the European Commission (AAL 2011-4-124) under the AAL Joint Program through the project "ALMA: Aging without Losing Mobility and Autonomy".

Appendix. Analysis of the orientation of the feedback-linearized system

In this appendix we analyze the behavior of the feedback-linearized unicycle model. In particular, we show that the wheelchair automatically aligns itself to the motion of point P, when point P is forced to follow a uniform motion in a straight line. This excludes unnatural movements of the wheelchair (e.g., backward motion). The dynamics of system (1), when the feedback-linearing control law (3) is applied, is

$$\begin{cases} \dot{x}(t) = (\cos\,\theta(t)v_{\mathrm{P}x}(t) + \sin\,\theta(t)v_{\mathrm{P}y}(t))\ \cos\theta(t) \\ \dot{y}(t) = (\cos\,\theta(t)v_{\mathrm{P}x}(t) + \sin\,\theta(t)v_{\mathrm{P}y}(t))\ \sin\theta(t) \\ \dot{\theta}(t) = \dfrac{1}{\varepsilon}(\cos\,\theta(t)v_{\mathrm{P}y}(t) - \sin\,\theta(t)v_{\mathrm{P}x}(t)) \end{cases} \tag{A.1}$$

Note that the point P displays a uniform motion in a straight line when inputs are $v_{\mathrm{P}x}(t) = v_{\mathrm{P}}\cos\theta_{\mathrm{P}}$ and $v_{\mathrm{P}y}(t) = v_{\mathrm{P}}\sin\theta_{\mathrm{P}}$, where $v_{\mathrm{P}} > 0$ and θ_{P} are constant in time. In this way we rewrite (A.1) as

$$\begin{cases} \dot{x}(t) = v_{\mathrm{P}}(\cos\,\theta(t)\cos\theta_{\mathrm{P}} + \sin\theta(t)\sin\theta_{\mathrm{P}})\ \cos\theta(t) \\ \dot{y}(t) = v_{\mathrm{P}}(\cos\theta(t)\cos\theta_{\mathrm{P}} + \sin\,\theta(t)\sin\theta_{\mathrm{P}})\ \sin\theta(t) \\ \dot{\theta}(t) = \dfrac{v_{\mathrm{P}}}{\varepsilon}(\cos\,\theta(t)\sin\theta_{\mathrm{P}} - \sin\,\theta(t)\cos\theta_{\mathrm{P}}) = -\dfrac{v_{\mathrm{P}}}{\varepsilon}\sin(\theta(t) - \theta_{\mathrm{P}}) \end{cases} \tag{A.2}$$

Defining $\Delta\theta(t) = \theta(t) - \theta_{\mathrm{P}}$, the third equation of (A.2) can be rewritten as

$$\Delta\dot{\theta}(t) = -\frac{v_{\mathrm{P}}}{\varepsilon}\sin\Delta\theta(t)$$

This implies that

- the equilibria $\theta = \theta_{\mathrm{P}} + 2k\pi$ (where $k \in \mathbb{Z}$) are asymptotically stable, with a basin of attraction corresponding to the open interval $(\theta_{\mathrm{P}} + (2k-1)\pi, \theta_{\mathrm{P}} + (2k+1)\pi)$;
- the equilibria $\theta = \theta_{\mathrm{P}} + (2k+1)\pi$, $k \in \mathbb{Z}$, are unstable, practically excluding backward motion.

References

Alessandretti, A., Aguiar, A. P., & Jones, C. N. (2013). Trajectory-tracking and path-following controllers for constrained underactuated vehicles using model predictive control. In *2013 European control conference* (pp. 1371–1376).

Alexis, K., Nikolakopoulos, G., & Tzes, A. (2012). Model predictive quadrotor control: attitude, altitude and position experimental studies. *IET Control Theory and Applications*, *6*(12), 1812–1827.

Bahadorian, M., Savkovic, B., Eaton, R., & Hesketh, T. (2012). Robust model predictive control for automated trajectory tracking of an unmanned ground vehicle. In *2012 American control conference* (pp. 4251–4256).

Bernardini, D., Cairano, S. D., Bemporad, A., & Tseng, H. E. (2009). Drive-by-wire vehicle stabilization and yaw regulation: a hybrid model predictive control design. In *2009 IEEE conference on decision and control* (pp. 7621–7626).

British-Standard-Institution, (1987). BS 6841:1987 –Guide to measurement and evaluation of human exposure to whole-body mechanical vibration and repeated shock.

Buizza Avanzini, G., Zanchettin, A. M., & Rocco, P. (2015). Constraint-based model predictive control for holonomic mobile manipulators. In *2015 IEEE international conference on intelligent robots and systems* (pp. 1473–1479).

Cavanini, L., Benetazzo, F., Freddi, A., Longhi, S., & Monteriù, A. (2014). SLAM-based autonomous wheelchair navigation system for AAL scenarios. In *IEEE/ASME international conference on mechatronic and embedded systems and applications* (pp. 1–5).

Celeste, W. C., Filho, T. F. B., Filho, M. S., & Carelli, R. (2008). Dynamic model and control structure for an autonomous wheelchair. In *2008 IEEE international symposium on industrial electronics* (pp. 1359–1364).

Ceravolo, E., & Gabellone, M. (Academic Year 2015–2016). *Controllo del moto di una sedia a rotelle autonoma mediante tecniche di controllo predittivo* (Master's thesis), Italy: Politecnico di Milano (in Italian).

Ceravolo, E., Gabellone, M., Farina, M., Bascetta, L., & Matteucci, M. (2017). Model predictive control of an autonomous wheelchair. In *2017 IFAC world congress* (pp. 9821–9826).

Cervantes, J., Yu, W., Salazar, S., Chairez, I., & Lozano, R. (2016). Output based backstepping control for trajectory tracking of an autonomous underwater vehicle. In *2016 American control conference* (pp. 6423–6428).

Chen, J., Sun, D., Yang, J., & Chen, H. (2010). Leader-follower formation control of multiple non-holonomic mobile robots incorporating a receding-horizon scheme. *International Journal of Robotics Research*, *29*(6), 727–747.

De Medio, C., Nicolò, F., & Oriolo, G. (1991). *Progress in systems and control theory*: *Vol. 7*. *Robot motion planning using vortex fields* (pp. 237–244).

De Nicolao, G., Ferrara, A., & Giacomini, L. (2007). Onboard sensor-based collision risk assessment to improve pedestrians' safety. *IEEE Transactions on Vehicular Technology*, *56*(5), 2405–2413.

Dunbar, W. B., & Murray, R. M. (2006). Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, *42*(4), 549–558.

Dydek, Z. T., Annaswamy, A. M., & Lavretsky, E. (2013). Adaptive control of quadrotor UAVs: A design trade study with flight evaluations. *IEEE Transactions on Control Systems Technology*, *21*(4), 1400–1406.

Falcone, P., Borrelli, F., Asgari, J., Tseng, H. E., & Hrovat, D. (2007). Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on Control Systems Technology*, *15*(3), 566–580.

Farina, M., Giulioni, L., & Scattolini, R. (2016). Stochastic linear model predictive control with chance constraints a review. *Journal of Process Control*, *44*(Supplement C), 53–67.

Farina, M., Perizzato, A., & Scattolini, R. (2015). Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots. *Robotics and Autonomous Systems*, *72*, 248–260.

Faulwasser, T., & Findeisen, R. (2016). Nonlinear model predictive control for constrained output path following. *IEEE Transactions on Automatic Control*, *61*(4), 1026–1039.

Fox, D., Burgard, W., & Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, *4*(1), 23–33.

Frasch, J. V., Gray, A., Zanon, M., Ferreau, H. J., Sager, S., & Borrelli, F. et al., (2013). An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles. In *2013 European control conference* (pp. 4136–4141).

Gao, Y., Gray, A., Carvalho, A., Tseng, H. E., & Borrelli, F. (2014). Robust nonlinear predictive control for semiautonomous ground vehicles. In *2014 American control conference* (pp. 4913–4918).

Gao, Y., Gray, A., Frasch, J. V., Lin, T., Tseng, H. E., & Hedrick, J. K. et al., (2012). Spatial predictive control for agile semi-autonomous ground vehicles. In *2012 international symposuim on advanced vechile control* (pp. 9–12).

Gao, Y., Lin, T., Borrelli, F., Tseng, H. E., & Hrovat, D. (2010). Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads. In *Dynamic systems and control conference*.

Gonzalez, R., Fiacchini, M., Guzman, J. L., & Alamo, T. (2009). Robust tube-based mpc for constrained mobile robots under slip conditions. In *2009 IEEE conference on decision and control* (pp. 5985–5990).

Gray, A., Gao, Y., Lin, T., Hedrick, J. K., Tseng, H. E., & Borrelli, F. (2012). Predictive control for agile semi-autonomous ground vehicles using motion primitives. In *2012 American control conference* (pp. 4239–4244).

Gu, D., & Hu, H. (2006). Receding horizon tracking control of wheeled mobile robots. *IEEE Transactions on Control Systems Technology*, *14*(4), 743–749.

Guo, J., Hu, P., & Wang, R. (2016). Nonlinear coordinated steering and braking control of vision-based autonomous vehicles in emergency obstacle avoidance. *IEEE Transactions on Intelligent Transportation Systems*, *17*(11), 3230–3240.

He, S., Zhang, R., Wang, Q., Chen, Y., Yang, T., Feng, Z., et al. (2017). A P300-based threshold-free brain switch and its application in wheelchair control. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *25*(6), 715–725.

Hu, C., Wang, R., Yan, F., & Chen, N. (2016). Output constraint control on path following of four-wheel independently actuated autonomous ground vehicles. *IEEE Transactions on Vehicular Technology*, *65*(6), 4033–4043.

International-Organization-for-Standardization, (1997). ISO 2631-1:1997 –Mechanical vibration and shock. Evaluation of human exposure to whole-body vibration General requirements.

Ji, J., Khajepour, A., Melek, W. W., & Huang, Y. (2017). Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. *IEEE Transactions on Vehicular Technology*, *66*(2), 952–964.

Katriniok, A., & Abel, D. (2011). LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics. In *2011 IEEE conference on decision and control and European control conference* (pp. 6828–6833).

Keviczky, T., Borrelli, F., Fregene, K., Godbole, D., & Balas, G. J. (2008). Decentralized receding horizon control and coordination of autonomous vehicle formations. *IEEE Transactions on Control Systems Technology*, *16*(1), 19–33.

Kühne, F., Lages, W. F., & Gomes Da Silva, J. M. (2004). Model predictive control of a mobile robot using linearization. In *2004 mechatronics & robotics conference* (pp. 525–530).

Kühne, F., Lages, W. F., & Gomes Da Silva, J. M. (2005). Point stabilization of mobile robots with nonlinear model predictive control. In *2005 IEEE international conference mechatronics and automation* (pp. 1163–1168).

Kuwata, Y., Richards, A., Schouwenaars, T., & How, J. P. (2007). Distributed robust receding horizon control for multivehicle guidance. *IEEE Transactions on Control Systems Technology*, *15*(4), 627–641.

Leaman, J., & La, H. M. (2017). A comprehensive review of smart wheelchairs: past, present, and future. *IEEE Transactions on Human-Machine Systems*, *47*(4), 486–499.

Lefèvre, S., Carvalho, A., & Borrelli, F. (2016). A learning-based framework for velocity control in autonomous driving. *IEEE Transactions on Automation Science and Engineering*, *13*(1), 32–42.

Lenz, D., Kessler, T., & Knoll, A. (2015). Stochastic model predictive controller with chance constraints for comfortable and safe driving behavior of autonomous vehicles. In *2015 IEEE intelligent vehicles symposium (IV)* (pp. 292–297).

Li, Z., Zhao, S., Duan, J., Su, C.-Y., Yang, C., & Zhao, X. (2017). Human cooperative wheelchair with brainmachine interaction based on shared control strategy. *IEEE/ASME Transactions on Mechatronics*, *22*(1), 185–195.

Liu, C., Wang, M., & Zhou, J. (2008). Coordinating control for an agricultural vehicle with individual wheel speeds and steering angles. *IEEE Control Systems*, *28*(5), 21–24.

Maniatopoulos, S., Panagou, D., & Kyriakopoulos, K. J. (2013). Model predictive control for the navigation of a nonholonomic vehicle with field-of-view constraints. In *2013 American control conference* (pp. 3967–3972).

Mayne, D. Q., Rawlings, J., Rao, C. V., & Scokaert, P. O. M. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, *36*(6), 789–814.

Megretski, A. (2003). Lecture notes in Dynamics of Non-Linear Systems. Technical Report.

Millán, P., Orihuela, L., Jurado, I., & Rubio, F. R. (2014). Formation control of autonomous underwater vehicles subject to communication delays. *IEEE Transactions on Control Systems Technology*, *22*(2), 770–777.

Oriolo, G., De Luca, A., & Vendittelli, M. (2002). WMR control via dynamic feedback linearization: design, implementation, and experimental validation. *IEEE Transactions on Control Systems Technology*, *10*(6), 835–852.

Palmieri, G., Barbarisi, O., Scala, S., & Glielmo, L. (2009). A preliminary study to integrate LTV-MPC lateral vehicle dynamics control with a slip control. In *2009 IEEE conference on decision and control* (pp. 4625–4630).

Parikh, S. P., Grassi, V., Kumar, V., & Okamoto, J. (2007). Integrating human inputs with autonomous behaviors on an intelligent wheelchair platform. *IEEE Intelligent Systems*, *22*(2), 33–41.

Plessen, M. G., Bernardini, D., Esen, H., & Bemporad, A. (2018). Spatial-based predictive control and geometric corridor planning for adaptive cruise control coupled with obstacle avoidance. *IEEE Transactions on Control Systems Technology*, *26*(1), 38–50.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., & Leibs, J. et al., (2009). ROS: An open-source robot operating system. In *2009 IEEE international conference on robotics and automation - open-source software workshop* (pp. 1–6).

Raffo, G. V., Gomes, G. K., Normey-Rico, J. E., Kelber, C. R., & Becker, L. B. (2009). A predictive controller for autonomous vehicle path tracking. *IEEE Transactions on Intelligent Transportation Systems*, *10*(1), 92–102.

Rasekhipour, Y., Khajepour, A., Chen, S. K., & Litkouhi, B. (2017). A potential field-based model predictive path-planning controller for autonomous road vehicles. *IEEE Transactions on Intelligent Transportation Systems*, *18*(5), 1255–1267.

Rawlings, J., & Mayne, D. (2009). *Model predictive control: theory and design*. Nob Hill Publishing LLC.

Rösmann, C., Feiten, W., Wösch, T., Hoffmann, F., & Bertram, T. (2012). Trajectory modification considering dynamic constraints of autonomous robots. In *2012 German conference on robotics* (pp. 74–79).

Rösmann, C., Feiten, W., Wösch, T., Hoffmann, F., & Bertram, T. (2013). Efficient trajectory optimization using a sparse model. In *2013 IEEE European conference on mobile robots* (pp. 138–143).

Rösmann, C., Hoffmann, F., & Bertram, T. (2015). Planning of multiple robot trajectories in distinctive topologies. In *2015 IEEE European conference on mobile robots* (pp. 1–6).

Ryll, M., Bülthoff, H. H., & Robuffo Giordano, P. (2015). A novel overactuated quadrotor unmanned aerial vehicle: modeling, control, and experimental validation. *IEEE Transactions on Control Systems Technology*, *23*(2), 540–556.

Sinyukov, D. A., & Padir, T. (2015). Adaptive motion control for a differentially driven semi-autonomous wheelchair platform. In *2015 international conference on advanced robotics* (pp. 288–294).

Truszkowski, W. F., Hinchey, M. G., Rash, J. L., & Rouff, C. A. (2006). Autonomous and autonomic systems: a paradigm for future space exploration missions. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *36*(3), 279–291.

Wang, R., Jing, H., Hu, C., Yan, F., & Chen, N. (2016). Robust H_∞ path following control for autonomous ground vehicles with delay and data dropout. *IEEE Transactions on Intelligent Transportation Systems*, *17*(7), 2042–2050.

Xie, F., & Fierro, R. (2007). On motion coordination of multiple vehicles with nonholonomic constraints. In *2007 American control conference* (pp. 1888–1893).

Xie, F., & Fierro, R. (2008). First-state contractive model predictive control of nonholonomic mobile robots. In *2008 American control conference* (pp. 3494–3499).

Zhang, R., Li, Y., Yan, Y., Zhang, H., Wu, S., Yu, T., et al. (2016). Control of a wheelchair in an indoor environment based on a braincomputer interface and automated navigation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *24*(1), 128–139.