Robotics and Autonomous Systems 58 (2010) 1–9

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

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

Optimal cooperative collision avoidance between multiple robots based on Bernstein–Bézier curves

Igor Škrjanc, Gregor Klančar [*]

Laboratory of Modelling, Simulation and Control, Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, SI-1000 Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 23 August 2007
Received in revised form
19 August 2009
Accepted 8 September 2009
Available online 15 September 2009

Keywords:
Mobile robots
Collision avoidance
Path planning

ABSTRACT

In this paper a new cooperative collision-avoidance method for multiple, nonholonomic robots based on Bernstein–Bézier curves is presented. The main contribution focuses on an optimal, cooperative, collision avoidance for a multi-robot system where the velocities and accelerations of the mobile robots are constrained and the start and the goal velocity are defined for each robot. The optimal path of each robot, from the start pose to the goal pose, is obtained by minimizing the penalty function, which takes into account the sum of all the path lengths subjected to the distances between the robots, which should be larger than the minimum distance defined as the safety distance, and subjected to the velocities and accelerations, which should be lower than the maximum allowed for each robot. The model-predictive trajectory tracking is used to drive the robots on the obtained reference paths. The results of the path planning, real experiments and some future work ideas are discussed.

1. Introduction

Collision avoidance is one of the main issues in applications for a wide variety of tasks in industry, human-supported activities, and elsewhere. Often, the required tasks cannot be carried out by a single robot, and in such a case multiple robots are used cooperatively. However, the use of multiple robots may lead to a collision if they are not properly navigated. Collision-avoidance techniques tend to be based on speed adaptation, route deviation by one vehicle only, route deviation by both vehicles, or a combined speed and route adjustment. When searching for the best solution to prevent a collision many different criteria are considered: time delay, total traveling distance or time, planned arrival time, etc. Our optimality criterion will be the minimum total traveling distance of all the mobile robots involved in the task, subject to a minimum safety distance between all the robots and subject to the velocity and acceleration constraints of each mobile robot. The approach can be adopted to include other utility functions as well as the limitations in the optimality criterion.

In the literature many different techniques for collision avoidance have been proposed. Some approaches proposed avoidance, when a collision between robots is predicted, by stopping the robots for a fixed period or by changing their directions. A combination of these techniques is proposed in [1,2]. The behavior-based motion planning of multiple mobile robots in a narrow passage is presented in [3]. Intelligent learning techniques were incorporated into neural and fuzzy control for mobile-robot navigation to avoid a collision, as proposed in [4,5]. A Bézier-curve-based path planning and collision avoidance for a single robot is presented in [6]. Also, some adaptive techniques for mobile robots' navigation have appeared, as proposed in [7].

In our case we are dealing with cooperative collision avoidance where all the robots are changing their paths cooperatively to achieve the goal. The Bézier-curve-based path planning is used, as in [6], but the optimal collision-safe trajectories are calculated for a group of robots at the same time and not for just a single robot, as in [6]. The collision-avoidance problem is formulated as an optimization problem, where the required safety distances, maximum velocities and accelerations of the mobile robots are constrained, and the start and the goal velocity are defined for each robot. This means that the proposed method can also be used as a subroutine in a huge path-planning problem where intermediate points are given along the searched path. The whole path is then composed of partial paths with smooth position and velocity transitions in connecting intermediate points. The control of multiple mobile robots to avoid collisions in a two-dimensional free-space environment is separated into two tasks: the path planning for each individual robot to reach its goal pose as fast as possible, and the trajectory-tracking control to follow the optimal path.

The trajectory-tracking task focuses on a controller design that will ensure the perfect trajectory tracking of the real mobile robots. Several controllers were proposed for mobile robots with nonholonomic constraints and an extensive review of nonholonomic

[*] Corresponding author. Tel.: +386 1 4768764; fax: +386 1 4264631.
E-mail address: gregor.klancar@fe.uni-lj.si (G. Klančar).

control problems can be found in [8]. In trajectory-tracking control a reference trajectory is usually obtained by using a reference robot; therefore, all the kinematics constraints are implicitly considered by a reference trajectory. From the reference trajectory a feed-forward system of inputs combined with a feedback control law are mostly used [9–11]. Lyapunov-stable, time-varying, state-tracking control laws were pioneered by [12,13]. Stabilization to the reference trajectory requires a nonzero motion condition. Many variations and improvements to this state-tracking controller followed in subsequent research [14,15]. for example a tracking controller obtained with input–output linearization is used in [9], a saturation feedback controller is proposed in [16] and a dynamic feedback linearization technique is used in [10].

In the field of mobile robotics predictive approaches to path tracking seem to be very promising, because the reference trajectory is known beforehand. However, the solution of the control problem is normally obtained by a minimization of some cost function. In [17] a generalized predictive control is chosen to control the mobile robot, minimizing the quadratic cost function. A generalized predictive controller using the Smith predictor to cope with an estimated system time delay is presented in [18]. In [19] a model-predictive control based on a linear, time-varying description of the system is used. The multi-layer, neural-network, predictive-controller scheme to a path-tracking problem is proposed in [20].

The proposed reference-tracking control is based on a prediction, where the main idea of the control law is to minimize the difference between a future trajectory following the errors of the robot and the reference path. The main advantage of the proposed predictive controllers is an explicitly obtained analytical control law that enables fast, real-time implementations.

The paper is organized as follows. In Section 2 the problem is stated. The concept of path planning is shown in Section 3. The idea of optimal collision avoidance for multiple mobile robots based on Bézier curves is discussed in Section 4. In Section 5 the proposed model-predictive controller is derived. The experimental results of the obtained collision-avoidance control are presented in Section 6 and the conclusion is given in Section 7.

2. Statement of the problem

The collision-avoidance control problem for multiple, nonholonomic mobile robots is proposed in a two-dimensional, free-space environment. Other static obstacles in the environment can be treated as motionless robots or as additional constraints in the optimization problem, which is presented in Section 4. The simulations and experiments were performed on small, two-wheel, differentially driven mobile robots with dimensions 7.5 × 7.5 × 7.5 cm. The architecture of our robots has a non-integrable constraint in the form $\dot{x}\sin\theta - \dot{y}\cos\theta = 0$, resulting from the assumption that the robot cannot slip in a lateral direction where $q(t) = [x(t)\ y(t)\ \theta(t)]^{\mathrm{T}}$ are the generalized coordinates, as defined in Fig. 1. The kinematics model of the mobile robot is

$$\dot{q}(t) = \begin{bmatrix} \cos\theta(t) & 0 \\ \sin\theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} \tag{1}$$

where $v(t)$ and $\omega(t)$ are the tangential and angular velocities of the platform. During low-level control the robot's velocities are bounded within the maximum allowed velocities, which prevents the robot from slipping.

The danger of a collision between multiple robots is avoided by determining the proper robot paths that fulfill certain criteria. The reference path of each robot, from the start pose to the goal pose, is obtained by minimizing the penalty function, which consists of the robots' traveling times as well as the required safety distance and the maximum velocity and acceleration constraints.

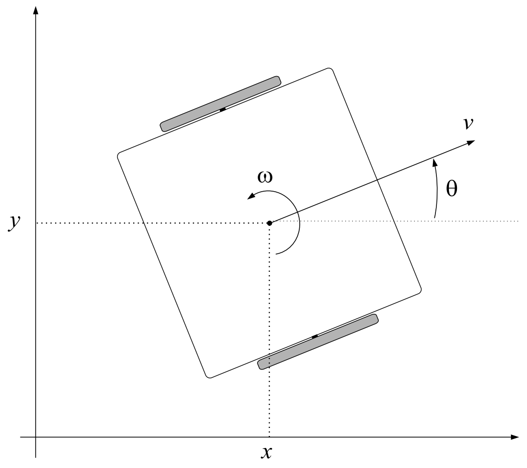

Fig. 1. The generalized coordinates of the mobile robot.

3. Path planning based on Bernstein–Bézier curves

Bézier curves were first published by Pierre Bézier in 1962 for designing the shape of automobile bodies. These parametric curves are a very important tool for modeling smooth curves in computer graphics and related fields as the curve is completely contained within the convex hull of its control points [21]. By performing operations on control points these curves can be translated and rotated.

In path-planning applications these curves are convenient because they can easily fit to the vehicle's boundary conditions (the start and end positions and the orientation) and can easily be derived to obtain the velocity and acceleration profile that is usually needed in path-tracking applications. The curvature of the Bézier curve varies smoothly from the starting point to the end point because of its continuous higher-order derivatives. The Bézier curve passes through the start and the final control point, but not through intermediate control points, which defines the start and the final orientation and the shape of the curve. The latter property is convenient in obstacle-avoidance applications.

Given a set of control points $P_0, P_1, \ldots, P_b$, the corresponding Bernstein–Bézier curve (or Bézier curve) is given by

$$\mathbf{r}(\lambda) = \sum_{i=0}^{b} B_{i,b}(\lambda)\mathbf{p}_i$$

where $B_{i,b}(\lambda)$ is a Bernstein polynomial, λ is a normalized time variable ($\lambda = t/T_{\max},\ 0 \le \lambda \le 1$) and $\mathbf{p}_i,\ 0 = 1, \ldots, b$ stands for the local vectors of the control point P_i ($\mathbf{p}_i = P_{i_x}\mathbf{e}_x + P_{i_y}\mathbf{e}_y$, where $P_i = \left(P_{i_x}, P_{i_y}\right)$ is the control point with the coordinates P_{i_x} and P_{i_y}, and $\mathbf{e}_x$ and $\mathbf{e}_y$ are the corresponding base unity vectors). The absolute maximum time $T_{\max}$ is the time needed to travel the path between the start control point and the goal control point. The Bernstein–Bézier polynomials, which are the base functions in the Bézier curve expansion, are given as follows:

$$B_{i,b}(\lambda) = \binom{b}{i} \lambda^i (1-\lambda)^{b-i}, \quad i = 0, 1, \ldots, b$$

and have the following properties: $0 \le B_{i,b}(\lambda) \le 1, 0 \le (\lambda) \le 1$ and $\sum_{i=0}^{b} B_{i,b} = 1$.

The Bézier curve always passes through the first and last control points and lies within the convex hull of the control points. The curve is at a tangent to the vector of the difference $\mathbf{p}_1 - \mathbf{p}_0$ at the start point and to the vector of the difference $\mathbf{p}_b - \mathbf{p}_{b-1}$ at the goal point [21]. A desirable property of these curves is that a curve can be translated and rotated by performing these operations on the control points. The undesirable properties of Bézier curves are their numerical instability for large numbers of control points, and the fact that moving a single control point changes the global shape of the curve. The former is sometimes avoided by smoothly patching together low-order Bézier curves.

The properties of Bézier curves are used in path planning for nonholonomic mobile robots. Their useful property is the tangentiality at the start and at the goal points, and the fact that moving a single control point changes the global shape of the curve. Let us assume the starting pose of the mobile robot is defined in the generalized coordinates as $\mathbf{q_0} = [x_0, y_0, \theta_0]^{\mathrm{T}}$ and the velocity in the start pose as v_0. The goal pose is defined as $\mathbf{q}_b = [x_b, y_b, \theta_b]^{\mathrm{T}}$ with the velocity in the goal pose as v_b, where b stands for the order of the Bézier curve. This means that the robot starts in position $P_0(x_0, y_0)$ with orientation θ_0 and velocity v_0 and has a goal defined by the position $P_b(x_b, y_b)$, the orientation θ_b and the velocity v_b.

Let us define five control points, P_0, P_1, P_2, P_3 and P_4, which uniformly define the fourth-order Bézier curve. The control points $P_1(x_1, y_1)$ and $P_3(x_3, y_3)$ are defined to fulfill the velocity and orientation requirements in the path. The need for flexibility of the global shape and the fact that moving a single control point changes the global shape of the curve imply the introduction of the control point denoted as $P_2(x_2, y_2)$. By changing the position of the point P_2 the global shape of the curve changes. This means that having in mind the flexibility of the global shape of the curve and the start and the goal pose of the mobile robot, the path can be planned by four fixed points and one variable point. The Bézier curve is now defined as the sequence of points P_0, P_1, P_2, P_3 and P_4 in Fig. 2, where D stands for the distance between the start and the goal control point. The Bernstein polynomials of the fourth order ($B_{i,b}$, $i = 0, \ldots, b$, $b = 4$), and the control points define the curve as follows:

$$\mathbf{r}(\lambda) = B_{0,4}\mathbf{p}_0 + B_{1,4}\mathbf{p}_1 + B_{2,4}\mathbf{p}_2 + B_{3,4}\mathbf{p}_3 + B_{4,4}\mathbf{p}_4 \tag{2}$$

or

$$\begin{aligned}\mathbf{r}(\lambda) = {} & (1-\lambda)^4 [x_0\ y_0]^{\mathrm{T}} + 4\lambda (1-\lambda)^3 [x_1\ y_1]^{\mathrm{T}} \\ & + 6\lambda^2 (1-\lambda)^2 [x_2\ y_2]^{\mathrm{T}} + 4\lambda^3 (1-\lambda) [x_3\ y_3]^{\mathrm{T}} \\ & + \lambda^4 [x_4\ y_4]^{\mathrm{T}}.\end{aligned} \tag{3}$$

The control point P_2 will be defined using the optimization, and the control points P_1 and P_3 are defined from the boundary velocity conditions. In the case of a large number of robots or obstacles an additional variable control point (or more) can be added to enable a more demanding curve shape with several turns. The number of control points can also be a variable, starting with a single one, and then increasing the number of them until progress is made in the optimization.

Let us therefore define the velocity as the derivation of the path vector $\mathbf{r}(\lambda)$ according to the normalized time λ as follows:

$$\mathbf{v}(\lambda) = \frac{\mathrm{d}\mathbf{r}(\lambda)}{\mathrm{d}\lambda} = \sum_{i=0}^{b-1} b \left(\mathbf{p}_{i+1} - \mathbf{p}_i\right) B_{b-1,i} \tag{4}$$

in the normalized time λ. In the case of the fourth-order ($b = 4$) curve, the velocity becomes:

$$\begin{aligned}\mathbf{v}(\lambda) = {} & 4(\mathbf{p}_1 - \mathbf{p}_0) B_{3,0} + 4(\mathbf{p}_2 - \mathbf{p}_1) B_{3,1} \\ & + 4(\mathbf{p}_3 - \mathbf{p}_2) B_{3,2} + 4(\mathbf{p}_4 - \mathbf{p}_3) B_{3,3}.\end{aligned} \tag{5}$$

The velocity vectors in the start position ($\lambda = 0$) and in the goal position ($\lambda = 1$) then become:

$$\begin{aligned}\mathbf{v}(0) &= 4\mathbf{p}_1 - 4\mathbf{p}_0 \\ \mathbf{v}(1) &= 4\mathbf{p}_4 - 4\mathbf{p}_3.\end{aligned} \tag{6}$$

This means that the vectors to the control points $\mathbf{p}_1$ and $\mathbf{p}_3$ are

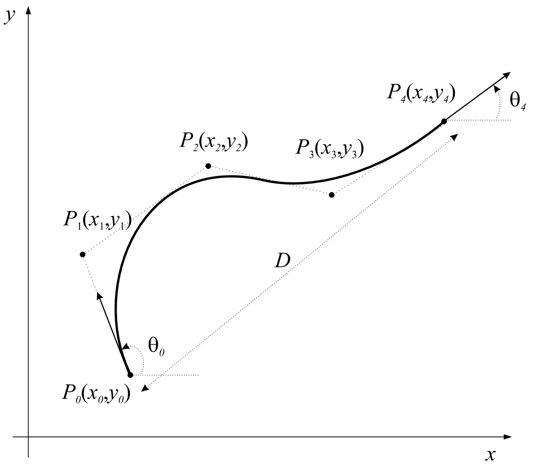

Fig. 2. The Bézier curve.

defined as follows:

$$\begin{aligned}\mathbf{p}_1 &= \mathbf{p}_0 + \frac{1}{4}\mathbf{v}(0) \\ \mathbf{p}_3 &= \mathbf{p}_4 - \frac{1}{4}\mathbf{v}(1).\end{aligned} \tag{7}$$

The goal velocity vector needs to be constrained if we want to arrive in the goal point with the required velocity. Moreover, the shape of path depends on the goal velocity (a higher velocity means a larger curve radius). If, however, the goal velocity is not prescribed, then the control point P_3 can be an additional curve-shape tuning parameter.

According to the orientation of the robot in the start and goal positions θ_s and θ_g, and given the start and required tangential velocities of the robot v_0 and v_4, the velocity vector can be written in the x and y components as follows:

$$\begin{aligned}\mathbf{v}(0) &= \left[v_x(0)\ v_y(0)\right]^{\mathrm{T}} = [v(0)\cos\theta_0\ v(0)\sin\theta_0]^{\mathrm{T}} \\ \mathbf{v}(1) &= \left[v_x(1)\ v_y(1)\right]^{\mathrm{T}} = [v(1)\cos\theta_4\ v(1)\sin\theta_4]^{\mathrm{T}}.\end{aligned} \tag{8}$$

Using Eqs. (7) and (8), the control points P_1 and P_3 are uniformly defined. The only unknown control point remains P_2, which will be defined by optimization to obtain the optimal path that will be collision-safe.

4. Optimal collision avoidance based on Bernstein–Bézier curves

In this subsection a detailed presentation of cooperative, multiple-robots, collision avoidance based on Bézier curves will be given, by taking into account the velocity constraints of the mobile robots. Let us assume the number of robots equals n. The ith robot is denoted as R_i and has a start position defined as P_{0i} (x_{0i}, y_{0i}) and a goal position defined as P_{4i} (x_{4i}, y_{4i}). The normalized time variable of the ith robot is denoted as $\lambda_i = t/T_{\max_i}$, where $T_{\max_i}$ stands for the absolute maximum time of the ith robot. The reference path will be denoted by the Bézier curve $\mathbf{r}_i(\lambda_i) = [x_i(\lambda_i), y_i(\lambda_i)]^{\mathrm{T}}$. In Fig. 3 a collision avoidance for $n = 2$ is presented for reasons of simplicity.

The safety margin to avoid a collision between two robots is, in this case, defined as the minimum necessary distance between these two robots. The distance between the robot R_i and R_j is $r_{ij}(t) = |\mathbf{r}_i(t) - \mathbf{r}_j(t)|, i = 1, \ldots, n, j = 1, \ldots, n, i \neq j$. By defining the minimum necessary safety distance as d_s, the following condition for collision avoidance is obtained: $r_{ij} \geq d_s,\ 0 \leq \lambda \leq 1,\ i, j$. Fulfilling this criterion means that the robots will never meet in the same region defined by a circle with radius

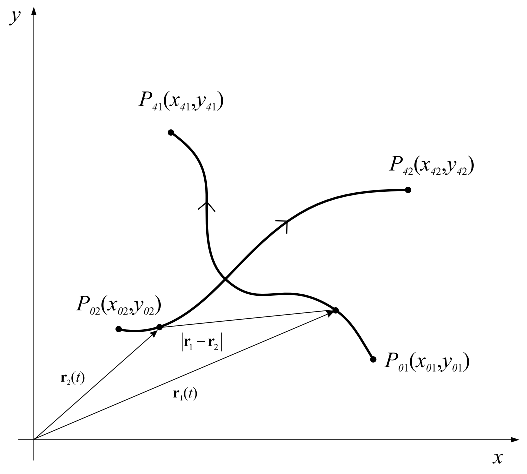

Fig. 3. Collision avoidance based on Bernstein–Bézier curves.

d_s, which is called a non-overlapping criterion. At the same time we would like to minimize the sum of the traveled paths s_i of all the robots. The length of the path at the normalized time $s_i(\lambda_i)$ is defined as $s_i(\lambda_i) = \int_0^{\lambda_i} v_i(\lambda_i)\mathrm{d}\lambda_i$, where $v_i(\lambda_i)$ stands for the tangential velocity of the ith robot in the normalized variable λ_i

$$v_i(\lambda_i) = |\dot{\mathbf{r}}(\lambda_i)| = \left(\dot{x}_i^2(\lambda_i) + \dot{y}_i^2(\lambda_i)\right)^{\frac{1}{2}}$$

where $\dot{x}_i(\lambda_i)$ stands for $\frac{\mathrm{d}x_i(\lambda_i)}{\mathrm{d}\lambda_i}$ and $\dot{y}_i(\lambda_i)$ for $\frac{\mathrm{d}y_i(\lambda_i)}{\mathrm{d}\lambda_i}$.

To define a feasible reference path that will be collision-safe and will satisfy the maximum velocity $v_{\max_i}$ and the maximum acceleration $a_{\max_i}$ of the mobile robot, the real time should be introduced. The relationship between the tangential velocity and acceleration in normalized time framework and the real tangential velocity and the acceleration is the following

$$v_i(t) = \frac{1}{T_{\max_i}} v_i(\lambda_i), \qquad a_i(t) = \frac{1}{T_{\max_i}^2} a_i(\lambda_i).$$

The length of the path of the robot R_i from the start control point to the goal point is now calculated as:

$$s_i = \int_0^1 \left((\dot{x}_i^2(\lambda_i)) + \dot{y}_i^2(\lambda_i)\right)^{\frac{1}{2}} \mathrm{d}\lambda_i$$

Assuming that the start P_{0i}, the goal P_{4i}, and the P_{1i} and P_{3i} control points are known, the global shape and length of each path can be optimized by changing the flexible control point P_{2i} and the maximum travel time $T_{\max_i}$ as well. By varying $T_{\max_i}$ the velocity profile of the planned path is changed accordingly. The collision-avoidance problem is now defined as an optimization problem, as follows:

$$\begin{aligned}
&\text{minimize} \sum_{i=1}^{n} s_i \\
&\text{subject to} \\
&d_s - r_{ij}(t) \le 0, \quad \forall i, j,\ i \ne j,\ 0 \le t \le \max_i(T_{\max_i}) \\
&v_i(t) - v_{\max_i} \le 0, \quad \forall i,\ 0 \le t \le \max_i(T_{\max_i}) \\
&a_i(t) - a_{\max_i} \le 0, \quad \forall i,\ \ 0 \le t \le \max_i(T_{\max_i}).
\end{aligned} \tag{9}$$

The minimization problem is called an *inequality optimization* problem. The methods using penalty functions transform a constrained problem into an unconstrained problem. The constraints are placed into the objective function via the penalty parameter in such a way as to penalize any violation of the constraints. In our case the following penalty function should be used to have an unconstrained optimization problem

$$\begin{aligned}
\text{minimize}_{(\mathbf{P}_2,\mathbf{T}_{\max})}\ F = &\sum_i s_i + c_1 \sum_{ij} \max_{ij}\left(0,\ 1/r_{ij}(t) - 1/d_s\right) \\
&+ c_2 \sum_i \max_i \left(0,\ v_i(t) - v_{\max_i}\right) \\
&+ c_3 \sum_i \max_i \left(0,\ a_i(t) - a_{\max_i}\right), \\
&i, j,\ i \ne j,\ 0 \le t \le \max_i(T_{\max_i})
\end{aligned} \tag{10}$$

where c_1, c_2 and c_3 stand for large scalars to penalize the violation of the constraints and the solution of the minimization problem $\min_{\mathbf{P}_2} F$ is a set of n control points $\mathbf{P}_2 = \{P_{21}, \dots, P_{2n}\}$ and $\mathbf{T}_{\max}$ is a set of n maximum times $\mathbf{T}_{\max} = \{T_{\max_1}, \dots, T_{\max_n}\}$. Choosing higher values of c_i increases the penalization of any constraint violation. If a particular c_i is higher than the others then this constraint has a higher priority. Each optimal control point $P_{2i},\ i = 1, \dots, n$ uniformly defines one optimal path, which ensures collision avoidance in the sense of a safety distance and will be used as a reference trajectory for the ith robot and will be denoted as $\mathbf{r}_i(\lambda)$. The optimal solution is also subjected to the time, because the velocities and accelerations of the robots are also taken into account in the penalty function (10).

5. Path tracking

The previously obtained optimal collision-avoidance path for the ith robot is defined as $\mathbf{r}_{ri}(t) = [x_{ri}(t), y_{ri}(t)]^{\mathrm{T}},\ i = 1, \dots, n$. In this section the development of a predictive path-tracking controller [22] is presented. The path-tracking control forces the robot to precisely follow the reference trajectory. Nevertheless, some small tracking errors may appear, which may in some extreme situations cause a violation of the safety distance constraint. Therefore, the penalty function (10) also includes constraints on the reference path velocity and the acceleration profile, which must always be lower than the robot's capabilities. So a planned reference path together with properly designed predictive control guarantees good trajectory-tracking performance. Considering these constraints the avoidance problem is decoupled, meaning that the trajectory planning is treated separately from the trajectory-tracking execution phase. This step significantly reduces the computational complexity of the path-planning optimization. Additionally, the safety distance can be extended to take a precaution. To improve the robustness of the path-tracking control in the case of large sensor noise, the safety distance could be extended according to the estimated sensor-noise standard deviation. In this work path-tracking control is realized as the sum of the feed-forward and feedback controls. The feed-forward control for the ith robot is calculated from a feasible reference path $\mathbf{r}_{ri}(t) = [x_{ri}(t), y_{ri}(t)]^{\mathrm{T}}$. The feed-forward control inputs $v_{ri}(t)$ and $\omega_{ri}(t)$ are derived using a kinematic model (1). The tangential velocity $v_{ri}(t)$ is calculated as follows

$$v_{ri}(t) = \left(\dot{x}_{ri}^2(t) + \dot{y}_{ri}^2(t)\right)^{\frac{1}{2}}. \tag{11}$$

The tangent angle of each point on the path is

$$\omega_{ri}(t) = \frac{\dot{x}_{ri}(t)\ddot{y}_{ri}(t) - \dot{y}_{ri}(t)\ddot{x}_{ri}(t)}{\dot{x}_{ri}^2(t) + \dot{y}_{ri}^2(t)} = v_{ri}(t)\kappa(t) \tag{12}$$

where $\kappa(t)$ is the path curvature. The necessary condition in the path-design procedure is a twice-differentiable path and a nonzero tangential velocity $v_{ri}(t) \ne 0$. If for some time t the tangential velocity is $v_{ri}(t) = 0$, the robot rotates at a fixed point with the angular velocity $\omega_{ri}(t)$ calculated from an explicitly given $\theta_{ri}(t)$.

The feedback control law is derived from a linear, time-varying system obtained by an approximate linearization around the trajectory. The obtained linearization is shown to be controllable

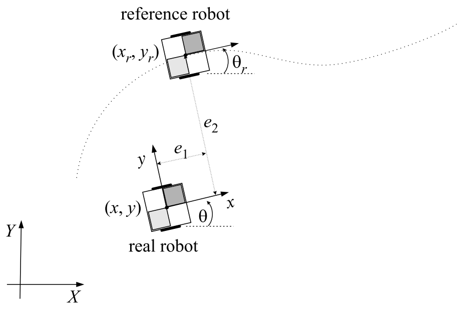

Fig. 4. Robot-following error transformation.

as long as the trajectory does not come to a stop, which implies that the system can be asymptotically stabilized by smooth, time-varying, linear or nonlinear feedback [13]. The tracking error $\mathbf{e}(t) = [e_1(t)\ e_2(t)\ e_3(t)]^{\mathrm{T}}$ of a mobile robot expressed in the frame of the real robot reads

$$\mathbf{e} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} (\mathbf{q}_{ri} - \mathbf{q}). \tag{13}$$

In Fig. 4 the reference robot ideally follows the reference path, but the real robot has some error when following the reference trajectory. Therefore, the control algorithm should be designed to force the robot to follow the reference path precisely.

Considering the robot kinematics (1) and the derived relations (13) the following kinematics model is obtained

$$\dot{\mathbf{e}} = \begin{bmatrix} \cos e_3 & 0 \\ \sin e_3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_{ri} \\ \omega_{ri} \end{bmatrix} + \begin{bmatrix} -1 & e_2 \\ 0 & -e_1 \\ 0 & -1 \end{bmatrix} \mathbf{u} \tag{14}$$

where $\mathbf{u} = [v\ \omega]^{\mathrm{T}}$ is the velocity input vector and v_{ri} and ω_{ri} are already defined in (11) and (12). The robot input vector $\mathbf{u}$ is further defined as the sum of the feed-forward and feedback control actions ($\mathbf{u} = \mathbf{u}_F + \mathbf{u}_B$) where the feed-forward input vector, $\mathbf{u}_F$, is obtained by a nonlinear transformation of the reference inputs $\mathbf{u}_F = [v_{ri}\cos e_3\ \ \omega_{ri}]^{\mathrm{T}}$ and the feedback input vector, is $\mathbf{u}_B = [u_{B_1}\ u_{B_2}]^{\mathrm{T}}$, which is the output of the controller defined in Section 5.1.

Using the relation $\mathbf{u} = \mathbf{u}_F + \mathbf{u}_B$ and rewriting (14) results in the following tracking-error model

$$\dot{\mathbf{e}} = \begin{bmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{e} + \begin{bmatrix} 0 \\ \sin e_3 \\ 0 \end{bmatrix} v_{ri} + \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{bmatrix} \mathbf{u}_B. \tag{15}$$

Furthermore, by linearizing the error dynamics (15) around the reference trajectory ($e_1 = e_2 = e_3 = 0, u_{B_1} = u_{B_2} = 0$) the following linear model results

$$\dot{\mathbf{e}} = \begin{bmatrix} 0 & \omega_{ri} & 0 \\ -\omega_{ri} & 0 & v_{ri} \\ 0 & 0 & 0 \end{bmatrix} \mathbf{e} + \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{bmatrix} \mathbf{u}_B \tag{16}$$

which in the state-space form is $\dot{\mathbf{e}} = \mathbf{A}_c\mathbf{e} + \mathbf{B}_c u_B$. According to Brockett's condition [23] a smooth stabilization of the system (1) or its linearization is only possible with time-varying feedback. In the following the obtained linear model is used in the derived predictive control law.

5.1. Model-predictive control based on a robot tracking-error model

To design the controller for trajectory tracking the system (16) will be written in discrete-time form as

$$\mathbf{e}(k+1) = \mathbf{A}\mathbf{e}(k) + \mathbf{B}\mathbf{u}_B(k)$$

where $\mathbf{A} \in \mathbb{R}^n \times \mathbb{R}^n$, n is the number of state variables, $\mathbf{B} \in \mathbb{R}^n \times \mathbb{R}^m$ and m is the number of input variables. The discrete matrix $\mathbf{A}$ and $\mathbf{B}$ can obtained as follows

$$\mathbf{A} = \mathbf{I} + \mathbf{A}_c T_s, \qquad \mathbf{B} = \mathbf{B}_c T_s \tag{17}$$

which is a good approximation during a short sampling time T_s.

The idea of the moving-horizon control concept is to find the control-variable values that minimize the receding-horizon, quadratic cost function (in a certain interval denoted with h) based on the predicted robot-following error:

$$J(u_B, k) = \sum_{i=1}^{h} \boldsymbol{\epsilon}^{\mathrm{T}}(k,i)\mathbf{Q}\boldsymbol{\epsilon}(k,i) + \mathbf{u}_B^{\mathrm{T}}(k,i)\mathbf{R}\mathbf{u}_B(k,i) \tag{18}$$

where $\boldsymbol{\epsilon}(k,i) = \mathbf{e}_{ri}(k+i) - \mathbf{e}(k+i|k)$ and $\mathbf{e}_{ri}(k+i)$ and $\mathbf{e}(k+i|k)$ stands for the reference robot-following trajectory and the robot-following error, respectively, and $\mathbf{Q}$ and $\mathbf{R}$ stand for the weighting matrices, where $\mathbf{Q} \in \mathbb{R}^n \times \mathbb{R}^n$ and $\mathbf{R} \in \mathbb{R}^m \times \mathbb{R}^m$, with $\mathbf{Q} \geq 0$ and $\mathbf{R} \geq 0$.

5.1.1. Output prediction in the discrete-time framework

In the moving time frame the model output prediction at the time instant h can be written as:

$$\begin{aligned} \mathbf{e}(k+h|k) &= \Pi_{j=1}^{h-1}\mathbf{A}(k+j|k)\mathbf{e}(k) \\ &\quad + \sum_{i=1}^{h}\left(\Pi_{j=i}^{h-1}\mathbf{A}(k+j|k)\right)\mathbf{B}(k+i-1|k)\mathbf{u}_B(k+i-1) \\ &\quad + \mathbf{B}(k+h-1|k)\mathbf{u}_B(k+h-1). \end{aligned} \tag{19}$$

Defining the robot-tracking, prediction-error vector

$$\mathbf{E}^*(k) = \left[e(k+1|k)^{\mathrm{T}}\ e(k+2|k)^{\mathrm{T}} \ldots e(k+h|k)^{\mathrm{T}}\right]^{\mathrm{T}}$$

where $\mathbf{E}^* \in \mathbb{R}^{n\cdot h}$ for the whole interval of observation (h) and the control vector

$$\mathbf{U}_B(k) = \left[\mathbf{u}_B^{\mathrm{T}}(k)\ \mathbf{u}_B^{\mathrm{T}}(k+1) \ldots \mathbf{u}_B^{\mathrm{T}}(k+h-1)\right]^{\mathrm{T}}$$

and

$$\boldsymbol{\Lambda}(k,i) = \Pi_{j=i}^{h-1}\mathbf{A}(k+j|k)$$

the robot-tracking, prediction-error vector is written in the form

$$\mathbf{E}^*(k) = \mathbf{F}(k)\mathbf{e}(k) + \mathbf{G}(k)\mathbf{U}_B(k) \tag{20}$$

where

$$\mathbf{F}(k) = \left[\mathbf{A}(k|k)\ \mathbf{A}(k+1|k)\mathbf{A}(k|k) \ldots \boldsymbol{\Lambda}(k,0)\right]^{\mathrm{T}}, \tag{21}$$

and

$$\mathbf{G}(k) = \begin{bmatrix} g_{11} & 0 & \cdots & 0 \\ g_{21} & g_{22} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ g_{n1} & g_{n2} & \cdots & g_{nh} \end{bmatrix} \tag{22}$$

$g_{11} = \mathbf{B}(k|k), \quad g_{21} = \mathbf{A}(k+1|k)\mathbf{B}(k|k)$
$g_{22} = \mathbf{B}(k+1|k), \quad g_{n1} = \boldsymbol{\Lambda}(k,1)\mathbf{B}(k|k)$
$g_{n2} = \boldsymbol{\Lambda}(k,2)\mathbf{B}(k+1|k), \quad g_{nh} = \mathbf{B}(k+h-1|k)$

and $\mathbf{F}(k) \in \mathbb{R}^{n\cdot h} \times \mathbb{R}^n$, $\mathbf{G}(k) \in \mathbb{R}^{n\cdot h} \times \mathbb{R}^{m\cdot h}$.

The objective of the control law is to drive the predicted robot trajectory as close as possible to the future reference trajectory, i.e., to track the reference trajectory. This implies that the future reference signal needs to be known. Let us define the reference error-tracking trajectory in state-space as

$$\mathbf{e}_{ri}(k+i) = \mathbf{A}_{ri}^{i}\mathbf{e}(k) \tag{23}$$

for $i = 1, \ldots, h$. This means that the future control error should decrease according to the dynamics defined by the reference model matrix $\mathbf{A}_{ri}$. Defining the robot reference-tracking-error vector

$\mathbf{E}^*_{ri}(k) = \left[\mathbf{e}_{ri}(k+1)^{\mathrm{T}}\ \mathbf{e}_{ri}(k+2)^{\mathrm{T}} \dots \mathbf{e}_{ri}(k+h)^{\mathrm{T}}\right]^{\mathrm{T}}$

where $\mathbf{E}^*_{ri} \in \mathbb{R}^{n\cdot h}$ for the whole interval of the observation (h) the following is obtained

$$\mathbf{E}^*_{ri}(k) = \mathbf{F}_{ri} e(k), \quad \mathbf{F}_{ri} = \left[\mathbf{A}_{ri}\ \mathbf{A}^2_{ri} \dots \mathbf{A}^h_{ri}\right]^{\mathrm{T}} \tag{24}$$

and $\mathbf{F}_{ri} \in \mathbb{R}^{n\cdot h} \times \mathbb{R}^n$.

5.1.2. Control law

The idea of MPC is to minimize the difference between the predicted robot-trajectory error and the reference robot-trajectory error in a certain predicted interval.

The cost function is, according to the above notation, now written as

$$J(U_B) = \left(\mathbf{E}^*_{ri} - \mathbf{E}^*\right)^{\mathrm{T}} \overline{\mathbf{Q}} \left(\mathbf{E}^*_{ri} - \mathbf{E}^*\right) + \mathbf{U}_B^{\mathrm{T}} \overline{\mathbf{R}} \mathbf{U}_B. \tag{25}$$

The control law is obtained by minimization ($\frac{\partial J}{\partial \mathbf{U}_B} = 0$) of the cost function and becomes

$$\mathbf{U}_B(k) = \left(\mathbf{G}^{\mathrm{T}} \overline{\mathbf{Q}} \mathbf{G} + \overline{\mathbf{R}}\right)^{-1} \mathbf{G}^{\mathrm{T}} \overline{\mathbf{Q}} \left(\mathbf{F}_{ri} - \mathbf{F}\right) \mathbf{e}(k) \tag{26}$$

where

$$\overline{\mathbf{Q}} = \begin{bmatrix} \mathbf{Q} & 0 & \cdots & 0 \\ 0 & \mathbf{Q} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{Q} \end{bmatrix}, \qquad \overline{\mathbf{R}} = \begin{bmatrix} \mathbf{R} & 0 & \cdots & 0 \\ 0 & \mathbf{R} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{R} \end{bmatrix}. \tag{27}$$

This means that $\overline{\mathbf{Q}} \in \mathbb{R}^{n\cdot h} \times \mathbb{R}^{n\cdot h}$ and $\overline{\mathbf{R}} \in \mathbb{R}^{m\cdot h} \times \mathbb{R}^{m\cdot h}$. Let us define the first m rows of the matrix $\left(\mathbf{G}^{\mathrm{T}} \overline{\mathbf{Q}} \mathbf{G} + \overline{\mathbf{R}}\right)^{-1} \mathbf{G}^{\mathrm{T}} \overline{\mathbf{Q}} \left(\mathbf{F}_{ri} - \mathbf{F}\right) \in \mathbb{R}^{m\cdot h} \times \mathbb{R}^n$ as $\mathbf{K}_{mpc}$. Now the feedback control law of the model-predictive control is given by

$$\mathbf{u}_B(k) = \mathbf{K}_{mpc} \cdot \mathbf{e}(k) \tag{28}$$

with $\mathbf{K}_{mpc} \in \mathbb{R}^m \times \mathbb{R}^n$.

6. Experimental results

In this section the path-planning results of the optimal, co-operative, collision-avoidance strategy between three real mobile robots is shown and the experimental results obtained on a real platform using model-predictive, trajectory-tracking control are given. The study was made to elaborate possible use in the case of a real mobile-robot platform. In a real platform we are faced with the limitation of control velocities and accelerations. Additional details about the real set-up and videos of the experiments are available at our website [24].

6.1. Case study for three mobile robots

The maximum allowed tangential velocities of the mobile robots are $v_{\max_i} = 0.8$ m/s and the maximum allowed accelerations are $a_{\max_i} = 0.5$ m/s^2, where $i = 1, 2, 3$.

The starting pose of the first mobile robot R_1 in generalized coordinates is defined as $\mathbf{q}_{01} = \left[0.2, 1.4, -\frac{\pi}{4}\right]^{\mathrm{T}}$ and the goal pose as $\mathbf{q}_{41} = \left[1.4, 0.2, -\frac{\pi}{4}\right]^{\mathrm{T}}$. The boundary velocities of the first mobile robot are the start tangential velocity $v_1(0) = 0.40$ m/s and the goal tangential velocity $v_1(T_{\max1}) = 0.4$ m/s. The second robot R_2 starts in $\mathbf{q}_{02} = \left[1.4, 0.2, \frac{3\pi}{4}\right]^{\mathrm{T}}$ and has the goal pose $\mathbf{q}_{42} = \left[0.2, 1.4, \frac{3\pi}{4}\right]^{\mathrm{T}}$. The boundary velocities of the second mobile robot are the start tangential velocity $v_2(0) = 0.4$ m/s and the goal tangential velocity $v_2(T_{\max2}) = 0.5$ m/s. The third

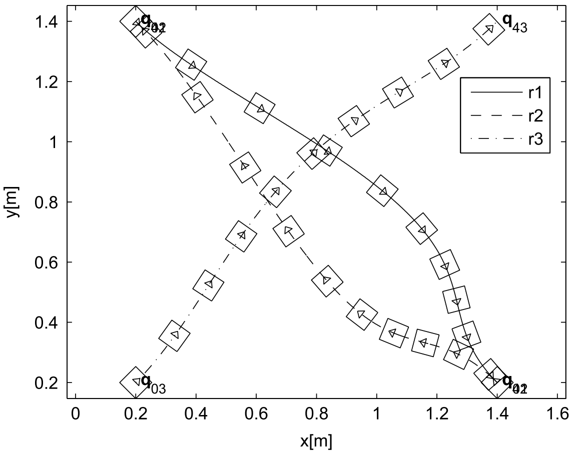

Fig. 5. The paths of the collision-avoiding robots R_1, R_2 and R_3.

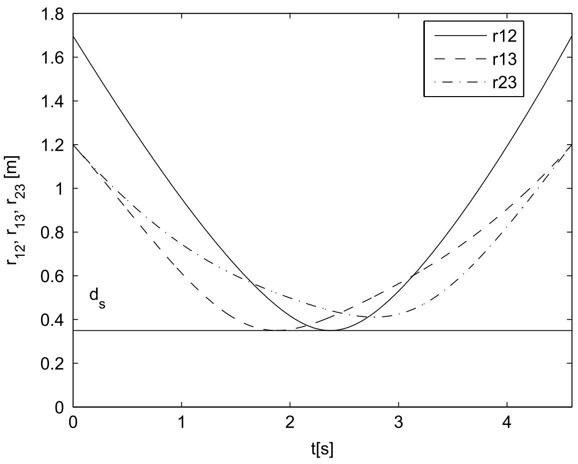

Fig. 6. The distances r_{12}, r_{13}, r_{23} between the robots R_1, R_2 and R_3.

robot R_3 starts in $\mathbf{q}_{03} = \left[0.2, 0.2, \frac{\pi}{4}\right]^{\mathrm{T}}$ and has the goal pose $\mathbf{q}_{43} = \left[1.4, 1.4, \frac{\pi}{4}\right]^{\mathrm{T}}$. The boundary velocities of the third mobile robot are the start tangential velocity $v_3(0) = 0.4$ m/s and the goal tangential velocity $v_3(T_{\max3}) = 0.4$ m/s. The x and y coordinates are defined in meters. The safety distance is defined as $d_s = 0.35$ m.

The optimal set $\mathbf{P}_2$ can be found by using one of the unconstrained optimization methods, but the initial conditions are very important. In these experiments the unconstrained, nonlinear, NelderMead simplex, direct-search, optimization method is used, which is implemented in the Matlab function fminsearch. The optimization should be started with initial parameters that ensure a feasible solution. We are optimizing the total sum of all the path lengths that are subjected to certain conditions based on the safety distances, velocities and accelerations of the robots. The velocity condition implies the implementation of the maximum time for each robot into the optimization routine. This implies that the initial set $\mathbf{P}_2$ will be defined as

$\mathbf{P}_2 = \{(x_{21}, y_{21}),\ (x_{22}, y_{22}),\ (x_{23}, y_{23})\}$

where x_{2i} and y_{2i} are defined as follows:

$$x_{2i} = \frac{x_{0i} + x_{4i}}{2}, \qquad y_{2i} = \frac{y_{0i} + y_{4i}}{2}, \quad i = 1, 2, 3. \tag{29}$$

The initial maximum times are defined as $T_{\max_i} = 5$ s ($i = 1, 2, 3$) to fulfill the maximum velocity constraints. The penalty function (10) parameters are $c_i = 100$ ($i = 1, 2, 3$). The obtained results of the optimization routine are the following $P_{21}(1.61, 0.70)$,

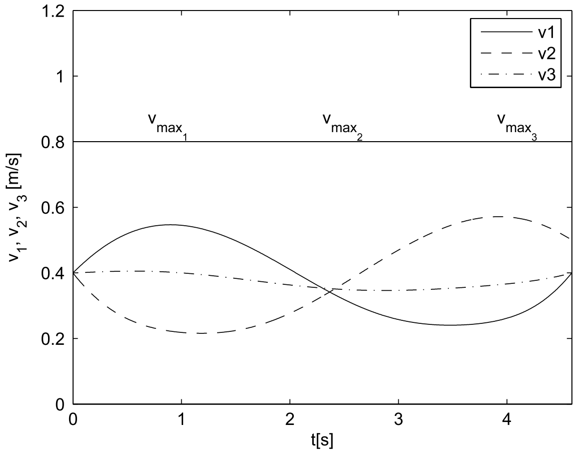

Fig. 7. The real velocities of the avoiding robots R_1, R_2 and R_3.

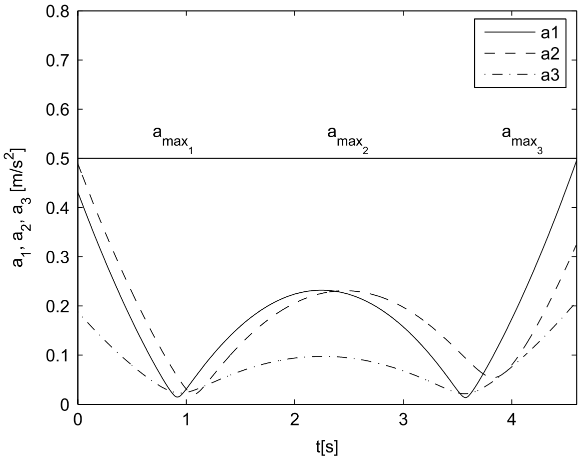

Fig. 8. The real accelerations of the avoiding robots R_1, R_2 and R_3.

$P_{22}(0.97, 0.02)$, $P_{23}(0.63, 1.10)$ and $T_{\max_1} = 4.5974$ s, $T_{\max_2} = 4.5973$ s and $T_{\max_3} = 4.5973$ s. The minimum value of the penalty function F is 5.2728.

The simulated positions of all three robots (R_1, R_2 and R_3) that are cooperatively avoiding a collision are shown in Fig. 5. Robot R_1 starts from the upper left corner and finishes in the lower right corner; robot R_2 starts from the lower right and finishes in the upper left corner; and robot R_3 starts from the lower left and finishes in the upper right corner. Obviously the paths of these robots cross. The robots (the square shapes on the trajectories in Fig. 5) are drawn each tenth sample time to illustrate their progress during the experiment. It is dear that the robots adjust their velocity profiles as well as their trajectories in order to fulfill the design constraints (d_s, $v_{\max_i}$ and $a_{\max_i}$).

In Fig. 6 the distances between the mobile robots are shown. It is also clear that all the distances (r_{12}, r_{13}, r_{23}) satisfy the safety distance condition. They are always larger than the prescribed safety distance d_s.

The real tangential velocity profiles of the avoiding robots R_1, R_2 and R_3 are given in Fig. 7. It is clear that the velocity profiles of all three robots fulfill the boundary velocity requirements and the allowed maximum velocity conditions. The acceleration profiles of the robots R_1, R_2 and R_3 are given in Fig. 8. All the accelerations fulfill the allowed maximum acceleration conditions.

In the following, Figs. 9–11, some more demanding scenarios are shown. In Fig. 9 the robots' starting poses are $\mathbf{q}_{01} = \left[0.4, 0.2, \frac{\pi}{2}\right]^{\mathrm{T}}$, $\mathbf{q}_{02} = \left[0.7, 0.2, \frac{\pi}{2}\right]^{\mathrm{T}}$ and $\mathbf{q}_{03} = \left[1, 0.2, \frac{\pi}{2}\right]^{\mathrm{T}}$ and the goal poses are $\mathbf{q}_{41} = \left[1.4, 1.4, \frac{\pi}{4}\right]^{\mathrm{T}}$, $\mathbf{q}_{42} = \left[1.4, 0.2, -\frac{\pi}{4}\right]^{\mathrm{T}}$ and $\mathbf{q}_{43} = \left[0.2, 1.4, \frac{3\pi}{4}\right]^{\mathrm{T}}$. All the robots have the same initial and final velocity, 0.25 m/s, while the required constraints, i.e., safety distance, maximum velocity and acceleration are $d_s = 0.25$ m, $v_{\max_i} = 0.8$ m/s and $a_{\max_i} = 0.5$ m/s^2, respectively.

Another scenario is shown in Fig. 10, where the robots' starting poses are $\mathbf{q}_{01} = \left[0.2, 0.6, \frac{\pi}{4}\right]^{\mathrm{T}}$, $\mathbf{q}_{02} = \left[0.4, 0.4, \frac{\pi}{4}\right]^{\mathrm{T}}$ and $\mathbf{q}_{03} = \left[0.6, 0.2, \frac{\pi}{4}\right]^{\mathrm{T}}$ and the goal poses are $\mathbf{q}_{41} = \left[1.4, 0.2, \frac{-\pi}{4}\right]^{\mathrm{T}}$, $\mathbf{q}_{42} = \left[1.4, 1.4, \frac{\pi}{4}\right]^{\mathrm{T}}$ and $\mathbf{q}_{43} = \left[0.2, 1.4, \frac{3\pi}{4}\right]^{\mathrm{T}}$. The required constraints, initial and final velocities are the same as in the experiment shown in Fig. 9.

If increasing the required safety distance in the experiment from Fig. 10 to $d_s = 0.28$ m (the maximum allowed for the initial robot poses) the obtained paths are shown in Fig. 11. The calculated paths are changed in order that they still obey the safety distance requirements.

In more general situations the proposed collision-avoidance algorithm always finds a solution that could be optimal or suboptimal, depending on how realistic the requirements and constraints are (safety distance, maximum allowed accelerations and velocities). To avoid local suboptimal solutions different initial sets for $\mathbf{P}_2$ need to be checked. An appropriate initial set for $\mathbf{P}_2$ can be obtained using relation (29). Nevertheless, the constrained problem is always feasible, as long as the design constraints are realistic.

As the proposed algorithm includes optimization it is computationally demanding and requires a fast computer to enable real-time operation, and it becomes even more computationally intense

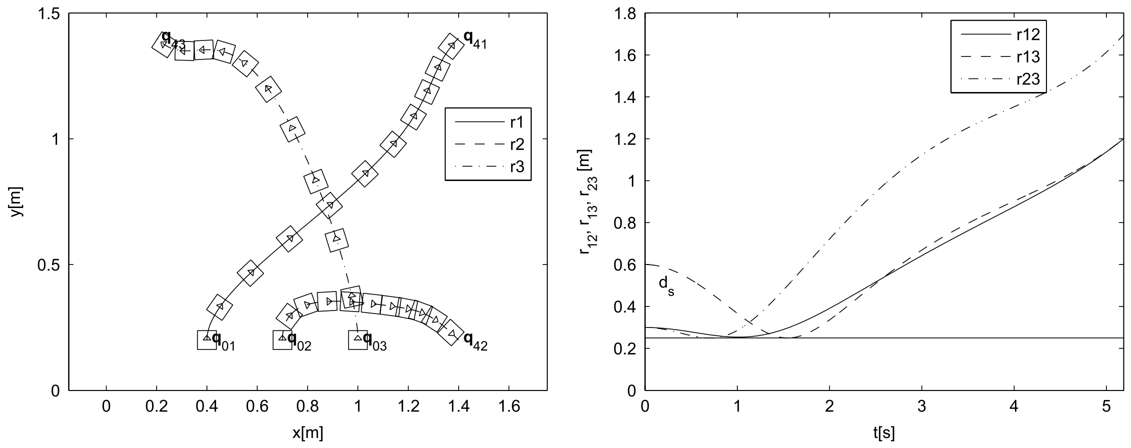

Fig. 9. The paths of the collision-avoiding robots R_1, R_2 and R_3 (left) and the distances r_{12}, r_{13}, r_{23} between them (right). The design constraints are $d_s = 0.25$ m, $v_{\max_i} = 0.8$ m/s and $a_{\max_i} = 0.5$ m/s^2.

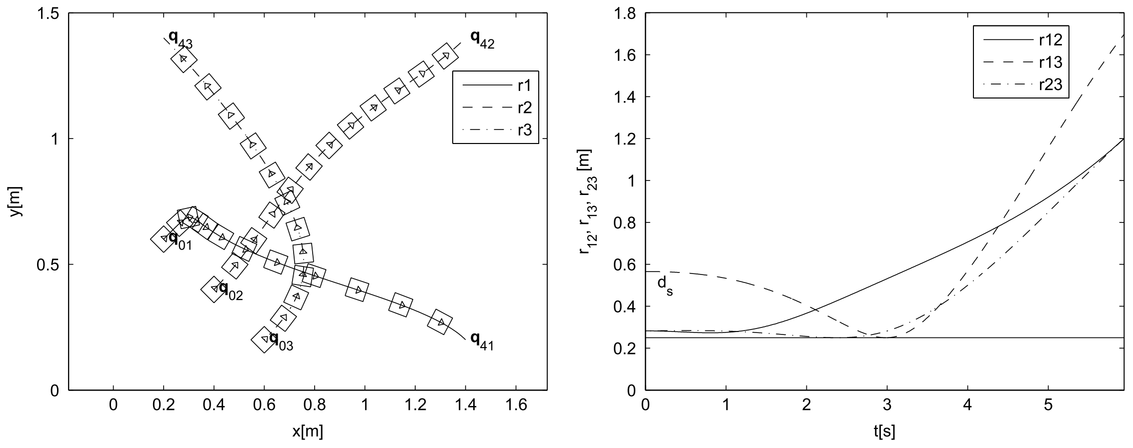

Fig. 10. The paths of the collision-avoiding robots R_1, R_2 and R_3 (left) and the distances r_{12}, r_{13}, r_{23} between them (right). The design constraints are $d_s = 0.25$ m, $v_{\max_i} = 0.8$ m/s and $a_{\max_i} = 0.5$ m/s^2.

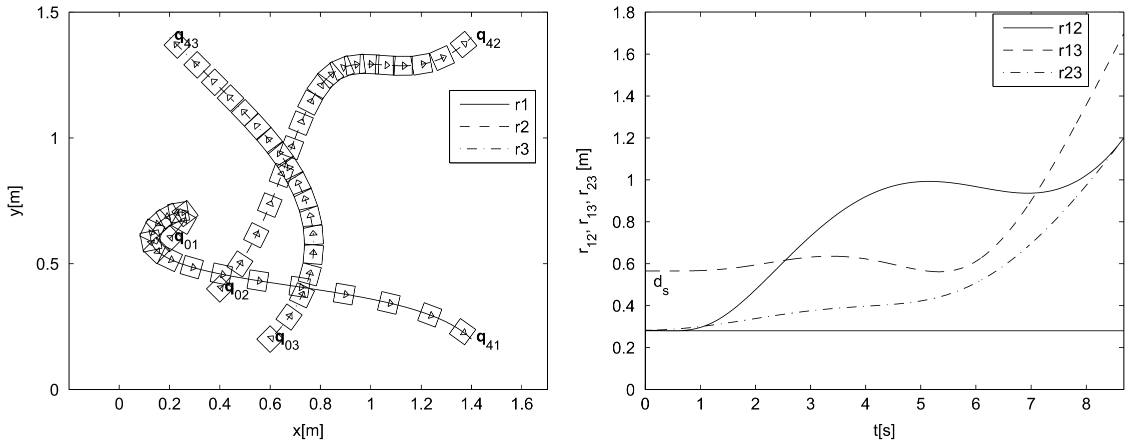

Fig. 11. The paths of collision-avoiding robots R_1, R_2 and R_3 (left) and distances r_{12}, r_{13}, r_{23} among them (right). Design constraints are $d_s = 0.28$ m, $v_{\max_i} = 0.8$ m/s and $a_{\max_i} = 0.5$ m/s^2.

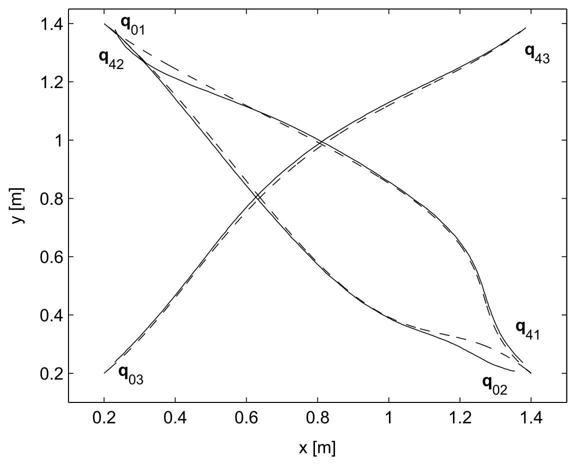

Fig. 12. The control of the collision-avoiding robots R_1, R_2 and R_3 (solid lines) on the reference trajectories (dashed lines); real experiment.

if the number of robots is increased. In the presented examples (Figs. 5–11) it took some 0.5 s to compute each experiment on a Pentium IV 1.8 GHz Computer in the Matlab environment.

In Fig. 12 the results of the experiment (from Fig. 5) performed on a small-sized real-robots platform (the size of each robot is 7.5 cm × 7.5 cm × 7.5 cm) is shown. It can be seen that the robots' initial postures $\mathbf{q}_{01}$, $\mathbf{q}_{02}$ and $\mathbf{q}_{03}$ have some initial pose error (they are not on the planned trajectory). These initial errors were introduced intentionally to demonstrate the operation of the designed predictive controller. The predictive controller successfully drives the robot to follow the reference trajectories, despite the noise in the position (standard deviation of 2 mm) and orientation measurements (standard deviation of 0.1 rad).

7. Conclusion

An optimal, cooperative, collision-avoidance approach based on Bézier curves allows us to include different criteria in the penalty functions. In our case the reference path of each robot, from the start pose to the goal pose, is obtained by minimizing the penalty function, which takes into account the sum of all the path lengths subjected to the distances between the robots, which should be larger than the minimum distance defined as the safety distance, the maximum velocities of the robots and the maximum allowed accelerations of the robots. The model-predictive trajectory-tracking control is used to control the robots on the obtained reference paths. The predictive control law minimizes the quadratic cost function consisting of tracking errors and control effort. The solution to the control is analytically derived, which enables fast, real-time implementations. Future improvements will focus on decreasing the computational time of the optimization by a problem reformulation or by coding in C.

The proposed, cooperative, collision-avoidance method for multiple nonholonomic robots based on Bézier curves and predictive, reference tracking shows great potential and in the future will be implemented on a real, large-scale, mobile-robot, Pioneer 3-AT platform.

Acknowledgment

The authors would like to acknowledge the Slovenian Research Agency for funding this work under the CRP MIR M2-0116 project ‘‘Mobile Robot System for Reconnaissance, Research and Rescue’’.

References

[1] R.C. Arkin, Cooperation without communication: Multiagent schema-based robot navigation, Journal of Robotic Systems 9 (3) (1992) 351–364.
[2] K. Sugihara, I. Suzuki, Distributed algorithms for formation of geometric patterns with many mobile robots, Journal of Robotic Systems 13 (3) (1996) 127–139.
[3] L. Shan, T. Hasegawa, Space reasoning from action observation for motion planning of multiple robots: Mutal collision avoidance in a narrow passage, Journal of Robot Society Japan 14 (1996) 1003–1009.
[4] C.G. Kim, M.M. Triverdi, A neuro-fuzzy controller for mobile robot navigation and multirobot convoying, IEEE Transaction on Systems, Man, and Cybernetics — Part B 28 (6) (1998) 829–840.
[5] N. Kubota, T. Morioka, F. Kojima, T. Fukuda, Adaptive behavior of mobile robot based on sensory network, JSME Transactions 65 (1999) 1006–1012.
[6] K. Jolly, R. Kumar, R. Vijayakumar, A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits, Robotics and Autonomous Systems 57 (2009) 23–33.
[7] A. Fujimori, P.N. Nikiforuk, M.M. Gupta, Adaptive navigation of mobile robots with obsticle avoidance, IEEE Transactions on Robotics and Automation 13 (4) (1999) 596–602.
[8] I. Kolmanovsky, N.H. McClamroch, Developments in nonholonomic control problems, IEEE Control Systems 15 (6) (1995) 20–36.
[9] N. Sarkar, X. Yun, V. Kumar, Control of mechanical systems with rolling constraints: Application to dynamic control of mobile robot, The International Journal of Robotic Research 13 (1) (1994) 55–69.
[10] G. Oriolo, A. Luca, M. Vandittelli, WMR control via dynamic feedback linearization: Design, implementation, and experimental validation, IEEE Transactions on Control Systems Technology 10 (6) (2002) 835–852.
[11] A. Balluchi, A. Bicchi, A. Balestrino, G. Casalino, Path tracking control for Dubin's cars, in: Proceedings of the 1996 IEEE International Conference on Robotics and Automation, Minneapolis, Minnesota, 1996, pp. 3123–3128.
[12] Y. Kanayama, Y. Kimura, F. Miyazaki, T. Noguchi, A stable tracking control method for an autonomous mobile robot, in: Proceedings of the 1990 IEEE International Conference on Robotics and Automation, Cincinnati, OH, vol. 1, 1990, pp. 384–389.
[13] C. Samson, Time-varying feedback stabilization of car like wheeled mobile robot, International Journal of Robotics Research 12 (1) (1993) 55–64.
[14] F. Pourboghrat, M.P. Karlsson, Adaptive control of dynamic mobile robots with nonholonomic constraints, Computers and Electrical Engineering 28 (4) (2002) 241–253.
[15] F.M. Raimondi, M. Melluso, A new robust fuzzy dynamics controller for autonomous vehicles with nonholonomic constraints, Robotics and Autonomous Systems 52 (2–3) (2005) 115–131.
[16] T.C. Lee, K.T. Song, C.H. Lee, C.C. Teng, Tracking control of unicycle-modeled mobile robots using a saturation feedback controller, IEEE Transactions on Control Systems Technology 9 (2) (2001) 305–318.
[17] A. Ollero, O. Amidi, Predictive path tracking of mobile robots, Application to the CMU Navlab, in: Proceedings of 5th International Conference on Advanced Robotics, Robots in Unstructured Environments, ICAR '91, Pisa, Italy, vol. 2,1991, pp. 1081–1086.
[18] J.E. Normey-Rico, J. Gomez-Ortega, E.F. Camacho, A Smith-predictor-based generalised predictive controller for mobile robot path-tracking, Control Engineering Practice 7 (6) (1999) 729–740.
[19] F. Kühne, J.M. Gomes da Silva Jr., W.F. Lages, Model predictive control of a mobile robot using linearization, in: Mechatronics and Robotics 2004, Aachen, Germany, 2004.
[20] D. Gu, H. Hu, Neural predictive control for a car-like mobile robot, Robotics and Autonomous Systems 39 (2) (2002) 73–86.
[21] E.W. Weisstein, Bézier curve, in: MathWorld — A Wolfram Web Resource, 2009. Available at: http://mathworld.wolfram.com/BezierCurve.html.
[22] G. Klančar, I. Škrjanc, Tracking-error model-based predictive control for mobile robots in real time, Robotics and Autonomous Systems 55 (6) (2007) 460–469.
[23] R.W. Brockett, Asymptotic stability and feedback stabilization, in: R.W. Brockett, R.S. Millman, H.J. Sussmann (Eds.), Differential Geometric Control Theory, Birkhuser, Boston, MA, 1983, pp. 181–191.
[24] G. Klančar, Optimal collision avoidance experiments, 2009. Available at: http://msc.fe.uni-lj.si/PublicWWW/Klancar/ColisionAvoidance.html.

Igor Škrjanc received the B.Sc., M.Sc., and Ph.D. degrees, all in electrical engineering, from the Faculty of Electrical and Computer Engineering, University of Ljubljana, Slovenia, in 1988, 1991, and 1996, respectively. He is currently an Associate Professor with the same faculty. His main research interests are in adaptive, predictive, fuzzy, and fuzzy adaptive control systems.

Gregor Klančar received his B.Sc. and Ph.D. degrees from the Faculty of Electrical Engineering of the University of Ljubljana, Slovenia, in 1999, and 2003, respectively. He is currently employed as an Assistant Professor with the same faculty. His research work focuses on the area of fault diagnosis methods, on the mobile robotics area and on the area of control and supervision of multiagent systems.