Analysis of path following and obstacle avoidance for multiple wheeled robots in a shared workspace

Article *in* Robotica · August 2018

DOI: 10.1017/S0263574718000875

Analysis of path-following and obstacle avoidance for multiple wheeled robots in a shared workspace

M. Hassan Tanveer, Carmine T. Recchiuto, Antonio Sgorbissa

Department of Informatics, Bioengineering, Robotics, and Systems Engineering (DIBRIS), University of Genova, Via Opera Pia 13, 16145, Italy

Abstract

The article presents the experimental evaluation of an integrated approach for path following and obstacle avoidance, implemented on wheeled robots. Wheeled robots are widely used in many different contexts, and they are usually required to operate in partial or total autonomy: in a wide range of situations, having the capability to follow a predetermined path and avoiding unexpected obstacles is extremely relevant. The basic requirement for an appropriate collision avoidance strategy is to sense or detect obstacles and make proper decisions when the obstacles are nearby. According to this rationale, the approach is based on the definition of the path to be followed as a curve on the plane expressed in its implicit form $f(x,y) = 0$, which is fed to a feedback controller for path following. Obstacles are modeled through Gaussian functions that modify the original function, generating a resulting safe path which – once again – is a curve on the plane expressed as $f'(x,y) = 0$: the deformed path can be fed to the same feedback controller, thus guaranteeing convergence to the path while avoiding all obstacles. The features and performance of the proposed algorithm are confirmed by experiments in a crowded area with multiple unicycle–like robots and moving persons.

Keywords: Wheeled robots, Path following, Moving Obstacle Avoidance

1. Introduction

An appropriate strategy for obstacle avoidance is a key factor for achieving safe navigation [1, 2, 3]. Many research works have focused on developing safe navigation algorithms in static environment [4, 5, 6, 7, 8]. However, in real world scenarios, assuming that the environment is static is unrealistic in most cases, and the environment should more realistically be modelled as dynamic: therefore, in the field of mobile robotics, the problem of safe navigation in dynamic environments is one of the most important challenges to be addressed [9, 10]. The challenge becomes more complex and tough when the information about the dynamic obstacles and the environment is not available, even if such information is often assumed to be present in many research works [11, 12, 13, 14, 15]. This information may include the complete map of the environments, the position and the orientation of the obstacles in the map, the nature of the obstacles (whether the shape of the obstacles is constant or varies over time) and the motion of the obstacles (whether the obstacle is moving with a constant or time-varying velocity).

The approach described in this article is based on a previously proposed framework for path-following [16], which introduces the idea of representing a path in 2D through the implicit equation of a curve in the plane $f(x,y) = 0$, and includes the definition of a feedback controller that takes the equation of the curve and the robot's pose to compute the path-following error. In [16] it has been formally demonstrated that the approach guarantees asymptotic convergence to the path (the approach has been extended, among the others, to N-trailers [17] and UAVs [18]).

Starting from previous work, the main contribution of the present article is to consider – for the first time in this framework – multiple dynamic obstacles during path following (e.g., other robots or persons). The article describes a formal procedure to model obstacles in such a way that collisions are guaranteed to be avoided, by producing a deformed path $f'(x,y) = 0$ that roughly follows the original one while avoiding all obstacles (the convergence to the deformed path being guaranteed, once again, by the feedback control law adopted [16]). The performance of the proposed approach in terms of closeness to the original path and farness from obstacles has been validated by quantitatively measuring its performance in a $3m \times 3m$ meters arena crowded with robots and persons. Even if the approach may partially resemble to Artificial Potential Fields or similar force field-based methods for navigation [19, 20], the presented approach has many peculiarities: among the others, it allows for setting the control variables (linear and angular speed) as a unique continuous function of the deformed path represented as a curve $f'(x,y) = 0$, the robot's pose x, y, θ, and the relative position x_j, y_j of all the locally-sensed obstacles with respect to the robot. The differences between the present approach and force-field methods are better discussed in Section 5.

The approach is applicable to path following and obstacle avoidance of unicycle–like mobile systems with bounded speed and angular velocity. It is well-known that the motion of many wheeled robots and unmanned aerial vehicles can be described by this model; see [21] and the references therein. A similar approach for modelling a path in presence of obstacles has been proposed in [22] for controlling a multicopter: however, the latter approach is different from the present article since it is not based on the feedback controller proposed in [16] and it has been only tested with a small number of static obstacles.

Section 2 describes the comparative analysis with the existing techniques. Section 3 describes materials and methods. Section 4 describes experiments performed with up to three robots and three persons moving within a $3m \times 3m$ arena. In Section 5, the results are discussed and conclusions are given.

2. Comparison with State-of-the-art

The problem of collision avoidance integrated with path following, in presence of both static and moving obstacles, has been widely studied and solutions have been put forward [23, 24, 25, 26, 27, 28]. While all the proposed solutions are able to generate a safe path (at least in presence of static obstacles), in order to evaluate the efficiency of a obstacle avoidance algorithm some additional aspects should be considered, e.g. computational cost, presence of limited or noisy information, necessity to estimate the position or velocity of moving obstacles and other robots, capability to find a path to the goal under any condition. In the following, some techniques in the Literature for path following and obstacle avoidance are explained and compared with the approach proposed in this article, taking into account their strengthness and weaknesses:

Model Predictive Control (MPC) is one of the most common solutions for obstacle avoidance [29]. Indeed, it has been applied to control various systems, including industrial systems[30, 31, 32, 33, 34, 35, 36, 37]. Moreover, it has also been used to generate safe trajectories for robots by using simplified dynamics in an unknown environment. An example of its application is the work of [38], in which Model Predictive Control (MPC) was applied for online avoidance of moving obstacles along with streamlined navigation towards the destination. In this framework, the controller predicts a future path and solves optimization problems to ensure collision-free trajectories. Variants of MPC have been proposed to allow mobile robots equipped with onboard sensors to avoid moving obstacles [39, 40, 41]. However, MPC is applicable for vehicles with simple linear models, while most vehicles exhibit more complicated non-holonomic characteristics with constraints on the linear and angular velocities. For this kind of vehicles, non-linear MPC is a more suitable control approach [42, 41]. In spite of its popularity, MPC requires prior knowledge of the robot model which increases the mathematical complexity: thus, the main drawback of this family of approaches is a significant computational burden associated with solving a set of nonlinear differential equations and a nonlinear dynamic optimization problem. On the contrary, the techniques proposed in this article requires very few computational resources, as a consequence of its simplicity.

Velocity Obstacle (VO) technique was first proposed in [43]. With some modifications, it is still extensively used in research works related to different domains [44, 45, 46]. To the end of motion planning, VO requires the set of all velocities of the robots and obstacles, assuming that both will maintain their current velocities. If a moving obstacle changes its velocity, then it could result in a collision, unless the path is not re-computed in real-time. The main disadvantage of this class of methods is that they take into account the obstacle velocities and that the robot behaviour does not change if the velocities of moving obstacles or other robots change (unless the path is periodically recomputed, which may be computationally expensive). Therefore, it is not well suited for highly dynamic scenarios. Also, estimating the velocities of moving objects and other robots using onboard sensors may be technically challenging. Our approach does not consider the velocity but only the position of other obstacles and robots (which is much easier to be estimated using onboard sensors), and periodically re-computes their position at high frequency. For this reason our approach may require obstacles and robots to move at a lower average velocity, but it is much less sensitive to sudden change in their velocity.

Artificial Potential Fields (APF) and their variants [19, 20] are still among the most widespread techniques for obstacle avoidance [47, 24, 48, 49, 50]. In APF, the robot is considered as a moving point in a potential field, where the goal generates attractive force and the obstacles produce repulsive forces: the method is very simple, and it can be straighforwardly applied to avoid moving obstacles, by knowing only their position relatively to the robot. However, it is a well-know drawback of APF that the robot may be trapped in a local minimum, thus preventing it to find a path to the goal. With the approach proposed in this article, it is guaranteed that by appropriately tuning the distance of influence of obstacles, the robot will be never trapped in local minima and a path toward the goal will be always found. This property of the algorithm is formally proven in [51]

The edge detection approach, with its more recent variants [52, 53], is also worth of mention. In this approach the robot takes into consideration the vertical edges of the obstacle; then, it looks for lines connecting edges, and considers them as the boundaries of the obstacles. As a result, it tries to move along the boundary. One of the main drawbacks of the method concerns its practical implementation: indeed, it is usually necessary for the robot to stop in front of the robot to acquire accurate measurements of edges, since sensor data must be very accurate in order for the algorithm to work in a proper manner. Errors in sensor readings can result in the distortion of the original shape of the obstacle and hence a misreading may lead to a collision. On the contrary, the approach proposed in this article consider all sensor readings as if they belonged to different obstacles: the approach guarantees not to collide with any of them, thus being more robust to sensor noise [51] and allowing for real-time path-generation and updating while the robot is moving.

The Vector Field Histogram (VFH) approach has been also extendedly applied to ground robots [54, 26]. In the VFH approach, the space around the robot is divided into sections of the same size, and every section has a value that represent the likelihood of the obstacle. The map is then translated into a polar histogram, that represents the space around the robot and the nearness of the obstacles. Finally, the robot direction is selected through heuristics, and can be straightforwardly applied to avoid both static and moving obstacles. The methods is suitable to work with sensors returning noisy information, however it has drawbacks similar to APF, in that - in complex environments - the method cannot guarantee that a path to the goal can be found even if it exists. With respect to VFH, the method proposed in this article has the same advantages that has been already mentioned when comparing it with APF [51].

Dynamic Window Approach (DWA), relies on the idea of performing a local search for admissible velocities that allow the robot to avoid obstacles while meeting kinematics constraints [55]. In order to reduce computational complexity, the search is performed within a dynamic window which is centred around the current velocities of the robot in the velocity space, and only circular curvatures are considered. A solution to avoid local minima is proposed in [56, 57] by introducing a planning stage in DW which produces collision-free local paths with a given velocity profile. Recently,[58] has proposed the Forbidden Velocity Map, a generalization of the Dynamic Window concept that considers the obstacle's and robot's shape, velocity and dynamics, to deal with navigation in unpredictable and cluttered scenarios. To take into account kinematics constraints, obstacle avoidance has been fully integrated with path following in [59] in which path following is achieved by controlling explicitly the rate of progression of a virtual target to be tracked along the path [60, 61], and obstacle avoidance relies on the deformable virtual zone principle, that defines a safety zone around the vehicle, in which the presence of an obstacle drives the vehicle reaction. However, as stated by authors, the combination of path following with a reactive obstacle avoidance strategy has a natural limitation coming from the situation where both controllers yield antagonist system reactions. This situation leads to a local minimum problem similar to APF, where a heuristic switch between controllers is necessary. The method proposed in this article includes an algorithm for path deformation in presence of obstacles and for path following, which guarantees at the same time goal-reachability [51] as well as Lyapunov convergence to the deformed path [16].

Other method for obstacle avoidance exist, typically based on graph-search algorithms: Simulated annealing [62], A* [63], Differential game approaches [64], Rapidly-exploring Random Trees [65, 66], etc. They have not been taken into account in this analysis because they assume that a map of the environment is a priori available, whereas the method proposed in this article is a local method with no global knowledge of the environment.

3. Materials and methods

In this work, a unicycle–like mobile robot has been considered as a case study. The kinematics of a unicycle–like robot comprises linear and angular motion which can be represented as:

$$\begin{aligned} \dot{x} &= u\cos\theta \\ \dot{y} &= u\sin\theta \\ \dot{\theta} &= r \end{aligned} \tag{1}$$

where x, y and θ correspond to the position and orientation of the robot with respect to a fixed frame, u is the linear velocity and r is the angular velocity (i.e., the control inputs).

3.1. Path Following

The control structure for the system described in this article is an extension of [16], which describes the implementation of the method for path following of wheeled robots: the aim of the current work is to evaluate the capabilities and performance of the control algorithm also in presence of fixed and moving obstacles. In the proposed method, the path to be followed is defined as a curve on the plane expressed in its implicit form $f(x,y) = 0$: Figure 1 shows this concept[1].

It may be noticed that the value of the function $f(x,y)$ while the robot is in position x, y represents the error from the path. Indeed, when the robot is following the curve (i.e., it is on the desired path) it holds $f(x,y) = 0$, whereas the value of $f(x,y)$ locally increases or decreases when the robot abandons the path.

[1]To be more precise, the path in Figure 1 is given by the intersection of a cylindrical surface in the 3D space $f(x,y,z) = 0$ with a plane: however, in the rest of this article, we assume that such intersection is produced with the XY plane (i.e., the plane described by the implicit equation $z = 0$). In this case, it is possible to make the z variable disappear, and represent the path through a single implicit equation $f(x,y) = 0$

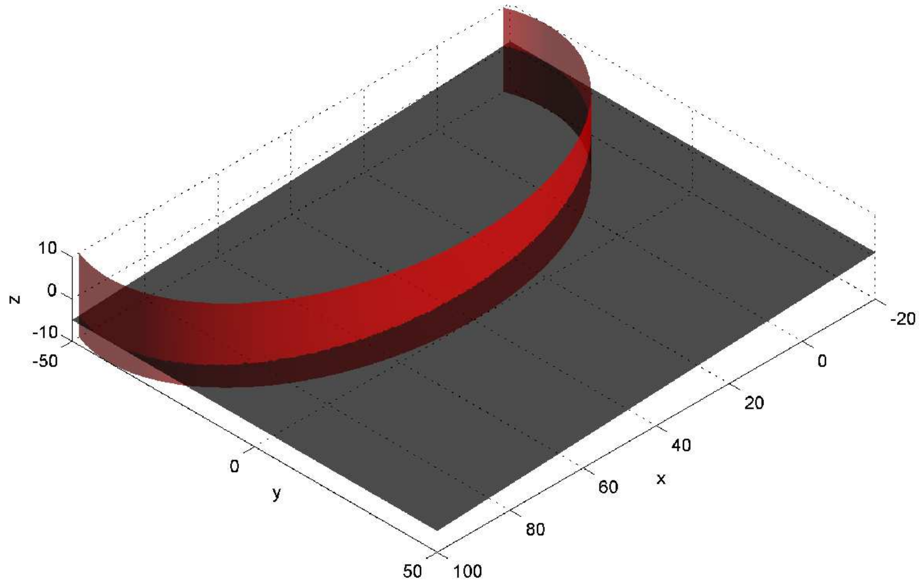

Figure 1: Path definition through surface intersection.

The function $f(x,y)$ must meet the following constraints:

(i) f has to be twice differentiable, with derivative f_x and f_y.

(ii) The norm of the gradient $||\nabla f||^2 = f_x^2 + f_y^2 > 0$.

Under these assumptions, it has been shown [16] that the robot can converge to the path by setting the control inputs as follows:

$$\begin{aligned} u &= u(t) \\ r &= K_1(-||\nabla f|| u S(f) - f_x |u| \cos\theta - f_y |u| \sin\theta) + \dot{\theta}_c \end{aligned} \tag{2}$$

$$\begin{aligned} S(f) &= \frac{K_2 f}{\sqrt{1+f^2}} \\ K_1, K_2 &> 0 \end{aligned} \tag{3}$$

where

describing a planar 2D curve.

- $u(t)$ is a positive velocity profile;
- K_1 and K_2 are gains;
- $\dot{f} = \dot{f}(x, y, \theta) = f_x u \cos\theta - f_y u \sin\theta$ describes how f varies with time, i.e., it is a measure of how fast the vehicle is getting closer to / farther from the path[2];
- $S(f)$ is the C^n sigmoid function, where K_2 determines the shape of the sigmoid;
- $\theta_c = \arg(f_y - i f_x)$ is the orientation of the vector $(f_y, -f_x)$ normal to ∇f in (x, y), i.e., tangent to the level curve, and $\dot{\theta}_c$ is its derivative with respect to time, which takes into account the curvature of the path.

The control law in (2) can be intuitively interpreted as follows. If the vehicle is in (x, y, θ) and it is moving along a level curve $w = f(x, y)$ with $w > 0$, it holds $\dot{f} = 0$ and $\dot{\theta} = \dot{\theta}_c$: in this case, the controller sets $\dot{\theta} = \dot{\theta}_c - K_1 \|\nabla f\| u S(w)$, and the vehicle approaches the path by leaving the level curve with $w > 0$ on its left side. This follows the fact that $\dot{\theta} < \dot{\theta}_c$ since the second term is negative, i.e., $\dot{\theta}$ is set to a lower value than required to move on the level curve. Symmetrically, when the vehicle is moving along a level curve with $w < 0$, the controller sets $\dot{\theta} = \dot{\theta}_c + K_1 \|\nabla f\| u S(w)$ since $S(-w) = S(w)$, and the vehicle tends to the path by leaving the level curve on its right side as $\dot{\theta} > \dot{\theta}_c$. For a more detailed analysis of the control law in (2) and (3) see [16], which contains also a formal prove of asymptotic convergence to the path and an experimental evaluation of the impact of control gains K_1 and K_2 on the robot's trajectory.

3.2. Obstacle Avoidance

In order to avoid obstacles while following (as closer as possible) the desired path, the path itself may be deformed when the robit perceives the presence of any obstacle.

[2]In [16], the absolute value of the velocity $|u|$ is used instead of u, which guarantees convergence to the path even when the vehicle is moving backward. Here we limit our analysis to positive values for sake of simplicity.

This is done by introducing a Gaussian function with radial simmetry in correspondence of each obstacle $\mathscr{O}_j$ as in (4)

$$O_j(x,y) = A_j e^{\frac{-(x-x_j)^2+(y-y_j)^2}{\sigma^2}} \tag{4}$$

where,

- x_j, y_j represent the position of the obstacle;
- A_j is the amplitude of the Gaussian curve;
- σ is the standard deviation.

The idea is to add the obstacle function O_j to the left term of the equation $f(x,y) = 0$ defining the path to be followed. Please notice that the Gaussian curve in (4) is one of the possible candidates as an obstacle function (a different bell–shaped function may be adopted as well), and that the behavior of the robot in proximity of the obstacle can be modified by tuning the parameters σ and A_j. Figure 2 illustrates these concepts: the path $f(x,y) = 0$ obtained as the intersection of a cylinder with a plane is deformed by the presence of an obstacle. The result is a path $f'(x,y) = 0$ that avoids the obstacle, while staying as closer as possible to the original path[3].

By tuning the value of σ and A_j, it is possible to generate a collision free trajectory even in the presence of multiple moving obstacles. In presence of more obstacles, it is necessary to sum up all the individual obstacle contributions:

$$f'(x,y) = f(x,y) + \sum_{j=1}^{N} O_j(x,y) = 0 \tag{5}$$

where each obstacle j is modeled by its position and dimensions that influence the parameters x_j, y_j σ and A_j of the obstacle function O_j.

[3]Once again, the Figure shows the intersection of a cylindrical surface $f(x,y,z) = 0$ with a generic plane in 3D, and obstacles j are consequently modelled as 3D Gaussians $O_j(x,y,z) = 0$. However, in the article we consider only the plane $z = 0$ to make the z variable disappear in all equations, thus finally yielding obstacle functions expressed as $O_j(x,y) = 0$ and a deformed path $f'(x,y) = 0$.

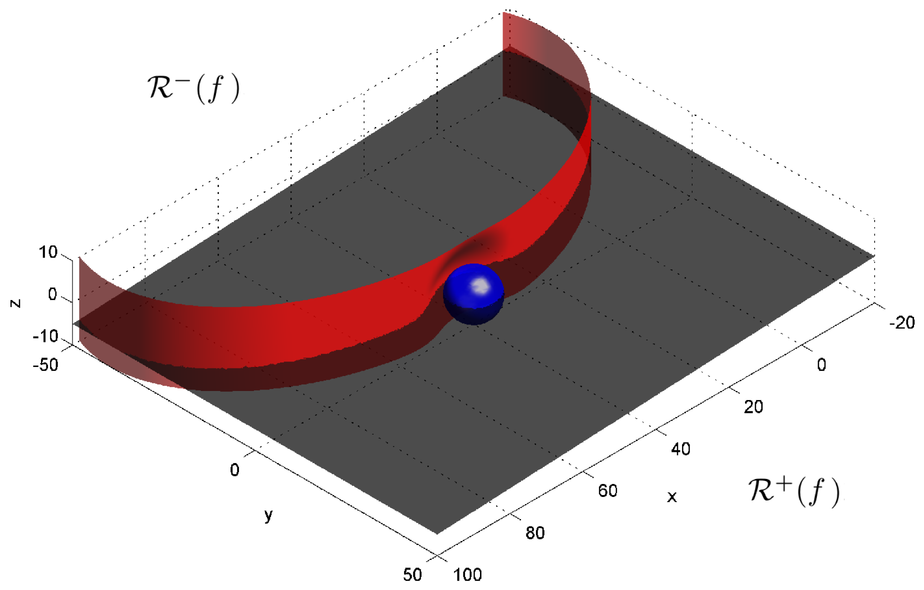

Figure 2: Deformation of the path due to the presence of an obstacle $\mathcal{O}_j$ with $A_j > 0$.

The obstacle detection and consequent path modification can be performed in real-time since it is sufficient to use the deformed path function $f'(x,y)$ instead of $f(x,y)$ in (2): as usual, to compute the path following error, $f'(x,y)$ needs to be evaluated only in the current robot's pose and its expression can be updated as soon as an obstacle has been detected. For instance, in presence of N obstacle and a nominal path corresponding – respectively – to a straight line $y = 0$, a circumference $x^2 + y^2 - R^2 = 0$, and a sine wave $y - sin(x) = 0$, the deformed path $f'(x,y) = 0$ would be computed as follow:

$$f'(x,y) = y + \sum_{j=1}^{N} O_j(x,y) = 0 \tag{6}$$

$$f'(x,y) = x^2 + y^2 - R^2 + \sum_{j=1}^{N} O_j(x,y) = 0 \tag{7}$$

$$f'(x,y) = y - sin(x) + \sum_{j=1}^{N} O_j(x,y) = 0. \tag{8}$$

3.3. Shaping the obstacle function to avoid collisions

The procedure to choose the values of σ and A_j deserves a deeper discussion.

First of all, it shall be noticed that the sign of A_j shall be chosen a priori. The undeformed curve $f(x,y) = 0$ divides the plane in two half-planes R^+ and R^-, and the deformed curve $f'(x,y) = 0$ lies on either R^+ or R^- depending on the sign of A_j: this determines whether obstacles are avoided on the right or on the left (Figure 2).

Second, to compute the absolute value of A_j, consider an obstacle $\mathscr{O}_j$ centered in a position x_j, y_j close to the path. A safety margin r_j shall be introduced to take into account both the dimensions of the obstacle and the vehicle: a collision may happen if the distance $d_j(x,y)$ between the position x,y of the vehicle and x_j, y_j is less or equal than the safety margin r_j. That is, $\mathscr{O}_j$ is defined as a circle

$$\mathscr{O}_j = \{(x,y) \; s.t. \; |(x,y) - (x_j,y_j)| \leq r_j\}. \tag{9}$$

In order to choose the actual value of A_j to avoid collisions, the path should not intersect any obstacle region.

Then, in presence of N obstacles, the following must hold:

$$f'(x,y) \neq 0, \; \forall (x,y) \in \bigcup_{j=1}^{N} \mathscr{O}_j \tag{10}$$

From (10) and (5) it follows that

$$f(x,y) + \sum_{j=1}^{N} O_j(x,y) \neq 0, \; \forall (x,y) \in \bigcup_{j=1}^{N} \mathscr{O}_j \tag{11}$$

which can be be solved for A_j by requiring either of the two situations below to hold:

$$f(x,y) + \sum_{j=1}^{N} O_j(x,y) > 0, \; \forall (x,y) \in \bigcup_{j=1}^{N} \mathscr{O}_j \tag{12}$$

or

$$f(x,y) + \sum_{j=1}^{N} O_j(x,y) < 0, \; \forall (x,y) \in \bigcup_{j=1}^{N} \mathscr{O}_j. \tag{13}$$

Suppose now that the arbitrary choice $A_j > 0$ has been made. In this case, it is convenient to satisfy (12), which allows us to simplify computations for the following reasons:

(i) the condition (12) is always satisfied for those obstacles $\mathscr{O}_j$ that lie completely in the semispace with $f(x_j,y_j) > 0$;

(ii) if the condition (12) is satisfied for each individual obstacle taken separately, it is also verified when considering all the obstacles.

Both properties above are due to the fact that, when $A_j > 0$, each individual obstacle adds a positive contribution in (12): this allows for computing A_j and σ to satisfy (12) for each obstacle taken separately.

Let us consider a generic obstacle $\mathscr{O}_j$: it can be observed that, since the Gaussian contribution in (4) has a radial simmetry, the minima of $O_j(x,y)$ in $\mathscr{O}_j$ necessarily lie on the boundary $\partial\mathscr{O}_j$, i.e., the circumference with radius r_j centered in x_j, y_j:

$$\min_{(x,y)\in\mathscr{O}_j} O_j(x,y) = A_j e^{-r_j^2/\sigma^2} \tag{14}$$

Moreover, in Section 3.1 we set the constraint $\nabla f(x,y) \neq 0$, with the effect that also the minima of $f(x,y)$ in $\mathscr{O}_j$ lie on the boundary $\partial\mathscr{O}_j$.

Then, from (12) it must hold

$$\min_{(x,y)\in\partial\mathscr{O}_j} f_1(x,y) + A_j e^{-r_j^2/\sigma^2} > 0, \tag{15}$$

hence

$$A_j > -\min_{(x,y)\in\partial\mathscr{O}_j} f(x,y) e^{r_j^2/\sigma^2}, \tag{16}$$

that yields, for each given σ, a lower bound on A_j.

Notice that, if we make the a priori choice $A_j < 0$, it is convenient to focus on (13), which guarantees the properties (i) and (ii) whereas (12) does not. After some computations, this finally requires to satisfy a condition similar to (16), but with the opposite inequality. Whichever choice is made for the sign of A_j, properties (i) and (ii) hold if and only if A_j has the same sign for all obstacles.

Lagrange multipliers can be used to find the minimum of $f(x,y)$ in $\partial\mathscr{O}_j$: this basically corresponds to finding the two level curves $f(x,y) = w_\alpha$ and $f(x,y) = w_\beta$ which are tangent to $\mathscr{O}_j$, respectively, in (x_α, y_α) and (x_β, y_β), and then taking the minimum between w_α and w_β.

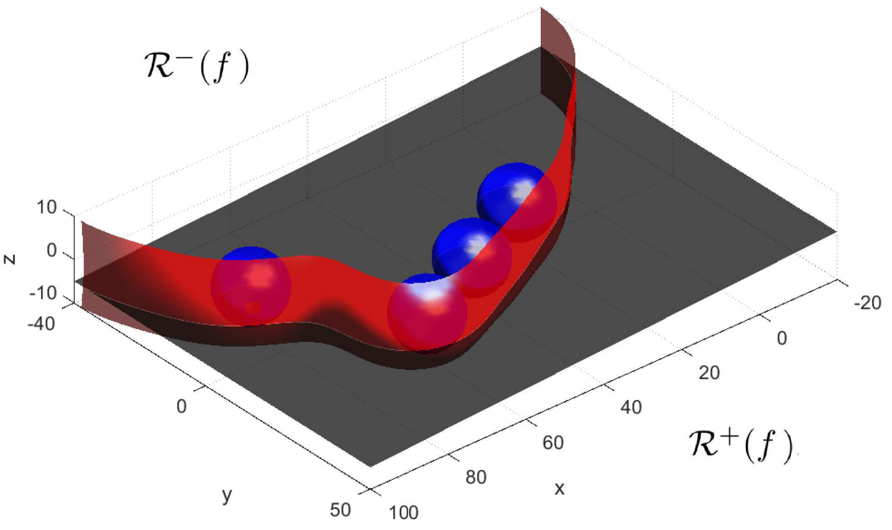

Figure 3: Deformation of the path due to the presence of multiple obstacles $\mathcal{O}_j$ with $A_j < 0$.

In the case that the initial path is a straight line $f(x,y) = ax + by + c$, after some computations it holds:

$$\begin{aligned} f(x_\alpha, y_\alpha) &= -\|\nabla f\| r_j + ax_j + by_j + c \\ f(x_\beta, y_\beta) &= \|\nabla f\| r_j + ax_j + by_j + c \end{aligned} \tag{17}$$

with the minimum corresponding to (x_α, y_α). Then, in order for the path not to intersect $\mathcal{O}_j$, the following relation must hold between A_j and σ:

$$A_j > -f(x_\alpha, y_\alpha) e^{r_j^2/\sigma^2}. \tag{18}$$

The procedure above must be reiterated for all obstacles $\mathcal{O}_j$ by substituting in (17), (18) the corresponding value of x_j, y_j. This allows for computing an admissible value for A_j depending on σ, thus guaranteeing that the deformed surface $f'(x,y) = 0$ does not collide with any obstacle, see Figure 3. In the case that the path $f(x,y) = 0$ is not a straight line, using the Lagrange multipliers to find $f(x_\alpha, y_\alpha)$ and $f(x_\beta, y_\beta)$ is not as computationally efficient as in the linear case. Therefore, a slightly different procedure is adopted which is based on the same rationale, but requires to approximate $f(x,y) = 0$ with a straight line (the whole procedure is not shown here for sake of brevity).

Finally notice that – depending on the value of σ and A_j – different paths are obtained: all of them guarantee that the constraint in (10) is met, but have different shapes. When σ is higher, the vehicle is influenced by obstacles at a greater distance, thus avoiding the obstacle along a lower curvature path.

3.4. Experimental setup

In order to evaluate the performance of the described approach, experiments have been performed with the robots Create, manufactured by iRobot. In particular, a variable number (from 1 to 3) of Create Robots move along predetermined and intersecting paths, so that the robots need to modify their paths in order to avoid colliding with each other. Additionally, a variable number of persons (from 1 to 3) are instructed to walk randomly in the same area, acting as mobile obstacles for the robots. All experiments have been performed within a $3m \times 3m$ area, inside a motion capture (MoCAP) environment (i.e: Motive Cap) that provides positioning feedback of any rigid body inside its perceptual field calculated by using 8 cameras located at the ceiling of the area. The system is based on the usage of reflective markers that are placed on the robots and the obstacles. The required 4 markers are placed on top of each robot and person's head as shown in Figure 4 and each set of markers is initialized as a rigid body with respect to the frame of reference. By measuring the size and the shape of the rigid body by using the markers, the system is able to precisely estimate the position and the orientation of wheeled robots and persons.

The whole architecture of the Robot includes Create and a processing board with Ubuntu Linux (version 12.04-2). Robots are wirelessly connected with a ground station, that receives feedback by the MoCap and works as a Master for all rigid bodies. The ROS (Robot Operating System) environment has been used in order to allow the robots and the ground station to communicate with each other. The positions of robots and obstacles in space are used as inputs for the proposed method, allowing the calculation of safe linear and angular velocities that can be generated through (2), (3).

Of course, more accurate systems or other additional velocity estimation algorithms based on pose information can be used [67]. The choice of a motion capture system instead of using on-board sensors for localization and obstacle detection was mainly

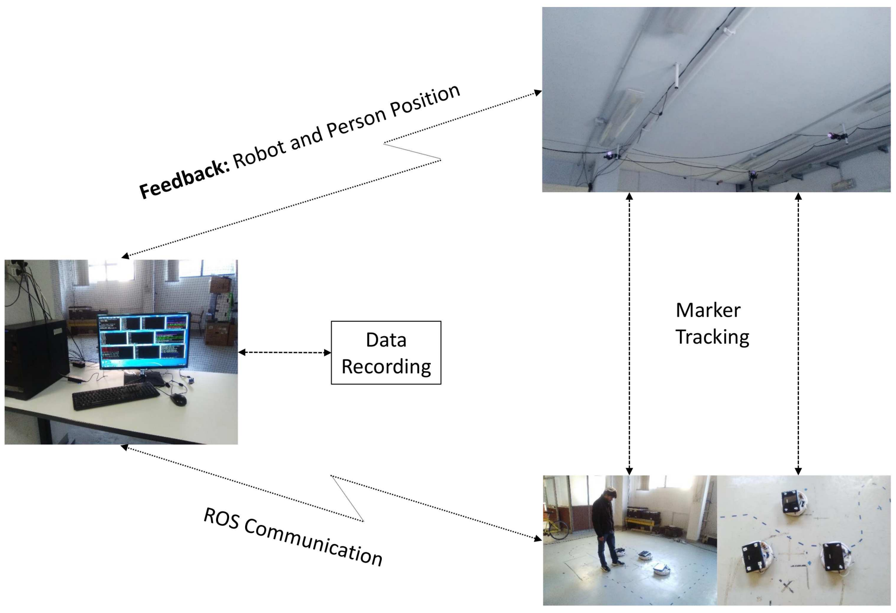

Figure 4: Structure of the system.

due to the fact that sensing and localization was not among the objectives of this work. Obviously, the method can be applied also by estimating the robot and obstacles position by using on-board sensors: in both cases, only relative pose information is required concerning the surrounding obstacles (including other robots), which makes the proposed method very effective when limited information is available.

In order to perform experiments, paths corresponding to a straight line, a sine wave and a circumference have been considered. Experiments have been grouped in 4 scenarios:

1. with no obstacles, to define a baseline to setup gains (A and σ) and compare the results with different speed.

2. with static obstacles, by changing the navigation speed from 0.1 to $0.6 m/s$ and with three different configurations:

    - One static obstacle;
    - Two obstacle placed together;

- Two obstacle placed apart from each other.

3. with moving obstacles:

   - circular path with two/three robots moving in the same area with different speed and gains;
   - sine wave, crossing, back and forth: with two robots moving along intersecting sine wave paths, by crossing each other's path;
   - sine wave, back and forth: with two robots moving along the same sine wave path, but starting in opposite direction.
   - straight line, crossing, back and forth: with two robots moving along intersecting straight line paths, by crossing each other's path.
   - straight line, back and forth: with two robots moving along the same straight line path, but starting in opposite direction.

4. with robots and persons, varing the number of persons from one to three.

The requirement for all cases is that the robots and persons should remain inside the predefined rectangular MoCap arena during the entire process. The results are shown in the following Figures and Tables. Notably, all the collision-avoidance cases designed are based on the assumption that the position of the static and moving obstacles in the environment can be carefully detected, therefore temporarily ignoring the problem of sensor noise. In order to interpret the results more appropriately, please consider the following additional information:

- A varying number of robot $k = 1....K$ has been considered, each moving along an actual path described as x_k, y_k.
- The equation used to calculate the error at time t of each robot k between its actual path and the desired (undeformed) path is:

$$e_k = f(x_k, y_k) \tag{19}$$

where x_k, y_k are the coordinate of the robot at time t. The error e_k should be interpreted as a measure of how far the robot is forced to deviate from the desired (undeformed) path due to the presence of surrounding obstacles.

- The average over time and standard deviation of $|e_k|$ are calculated and denoted by $av(|e_k|)$ and $std(|e_k|)$.

- The euclidean distance between two agents k and i at time t (k and i can either be robots or persons) is computed and denoted by $dist_{k,i}$. When $dist_{k,i}$ is greater than a safety distance d_{safe}, this means that there have been no collisions between agents (the Create robots used in experiments can be modelled as circles with radius $r_k = 0.17m$, and we model persons in the same way for the purpose of the present analysis: then we assume $d_{safe} = 0.34m$). The minimum distance $dist_{k,i}$ between agent k and i that has been reported during an experiment is denoted as $md_{k,i}$.

4. Experimental Results

4.1. Scenario 1: Without Obstacles

The first experiment is aimed at validating the approach in absence of obstacles. The robot has been given a circular path (7) of radius $R = 0.7m$. The response of the robot while following the reference path is shown in Figure 5.

4.2. Scenario 2: With Static Obstacles

The second experiment is performed by adding a static obstacle in a selected location along the reference path, while the robot moves along a circular path with radius $R = 0.9m$. The response of the robot is shown in Figure 6.

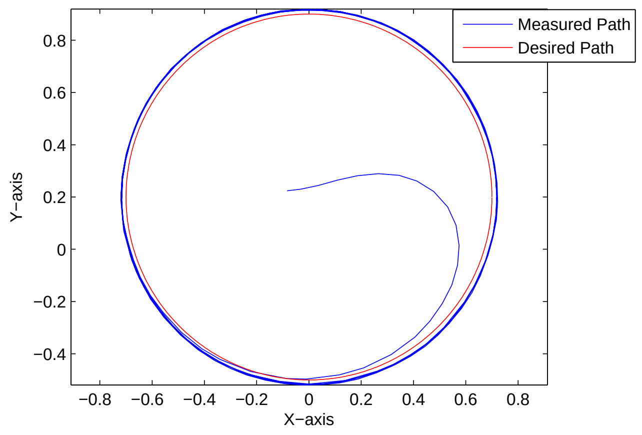

Figure 5: One Robot without obstacles: $u_1(t) = 0.3m/s, K_1 = 15, K_2 = 2$.

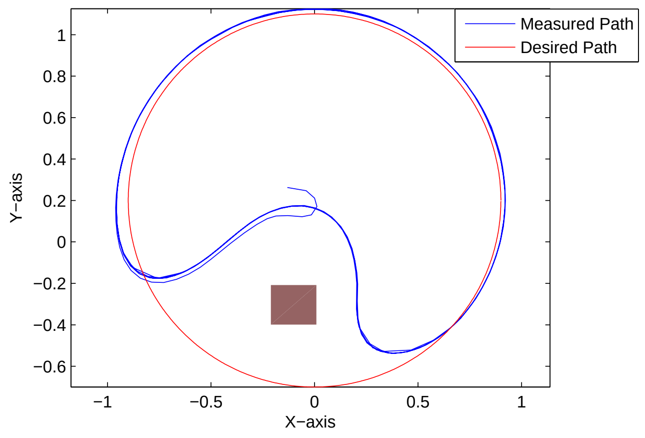

Figure 6: One Robot with One static obstacle: $u(t) = 0.3m/s, K_1 = 15, K_2 = 2, A_j = 0.8, \sigma = 0.5$.

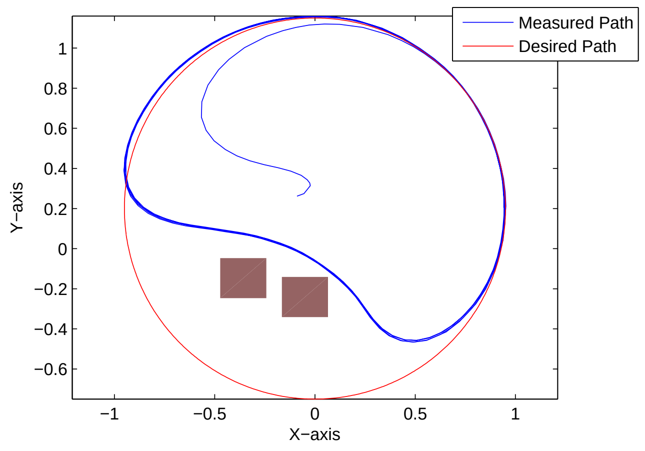

Figure 7: One Robot with Two static obstacles placed together:
$u(t) = 0.3m/s, K_1 = 15, K_2 = 2, A_j = 0.8, \sigma = 0.5$.

In the third case, two static obstacles are placed together. The robot response is shown in Figure 7. Indeed, the robot starts deviating earlier, but it remains closer to the predefined circular path.

In the last experiment of this scenario, static obstacles are placed along the path but apart from each other. The response in Figure 8 confirms the ability of the robot to follow the circular path by avoiding both obstacles.

Table 1 corresponds to the experiments performed in Scenarios 1 and 2 by varying the linear velocity from $0.1m/s$ to $0.6m/s$. The first column reports the velocity $u_1(t)$ of the robot (that is constant within each experimental run), the second, third, fourth and fifth columns report the average $av(|e_k|)$ and the standard deviation $std(|e_k|)$ of the error between the actual robot's path and the path to be followed, i.e., a measure of how far – on average – the robot is forced to deviate from the desired path $f(x,y) = 0$ due to the presence of surrounding obstacles (low values are an index of the fact that the robot tends to go back the path after avoiding obstacles). The sixth column reports the control gains that shall be properly tuned depending on the velocity. It can be noticed that, when moving in absence of obstacles along a circular path, the value of the linear velocity has no significant impact on of e_k when in the range $0.1m/s - 0.6m/s$.

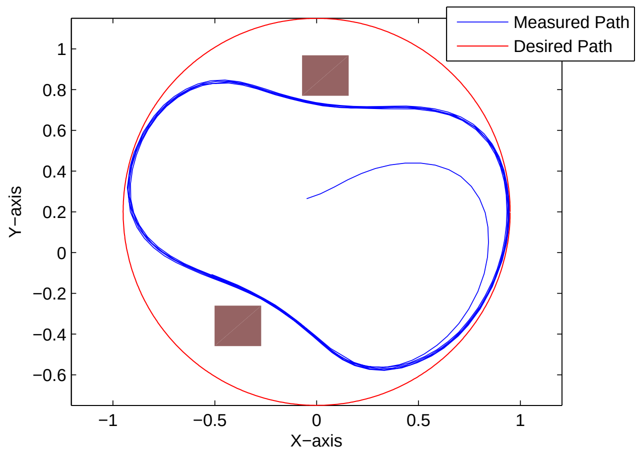

Figure 8: One Robot with Two obstacles placed apart
$u(t) = 0.3m/s, K_1 = 15, K_2 = 2, A_j = 0.8, \sigma = 0.5$.

Table 1: Summary: Robot's response by varying speed with and without fixed obstacles and $A_j = 0.8$, $\sigma = 0.5$.

	Without obstacles	One static obstacle	Two static obstacles together	Two static obstacles apart	
$u_1(t)$	$av(\|e_1)$, $std(\|e_1\|)$	$av(\|e_1)$, $std(\|e_1\|)$	$av(\|e_1)$, $std(\|e_1\|)$	$av(\|e_1)$, $std(\|e_1\|)$	K_1, K_2
0.1	0.038, 0.087	0.178, 0.242	0.186, 0.205	0.169, 0.195	20, 4
0.2	0.054, 0.076	0.201, 0.264	0.238, 0.211	0.143, 0.174	18, 3.5
0.3	0.034, 0.043	0.182, 0.263	0.165, 0.239	0.155, 0.156	15, 2
0.4	0.060, 0.088	0.210, 0.277	0.223, 0.283	0.168, 0.162	12, 1.6
0.5	0.048, 0.069	0.196, 0.264	0.213, 0.266	0.178, 0.167	11, 1.2
0.6	0.065, 0.092	0.187, 0.257	0.198, 0.244	0.162, 0.154	10, 1

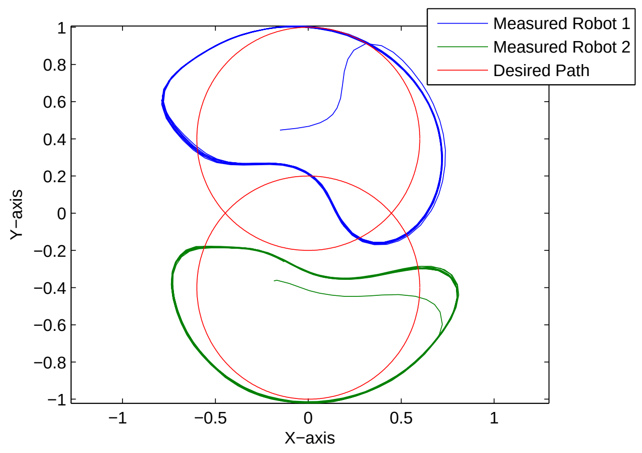

Figure 9: Two Robots with the same speed recognizing each other as obstacles: $u_1(t) = u_2(t) = 0.3m/s, K_1 = 15, K_2 = 2, A_j = 0.5, \sigma = 0.45$.

4.3. Scenario 3: With moving Obstacles

4.3.1. Circular Path

The third phase of experiments is performed with moving obstacles, i.e. more robots are placed in the arena, moving along their respective paths and avoiding each other. Indeed, since paths are intersecting each other, each robot recognizes the other robots as obstacles.

In the first test, two robots are used. The robots start moving with a distance of $0.8m$, and follow two circular paths with radius $R = 0.6m$ with the same speed $u_1(t) = u_2(t)$, see Figure 9. Figure 10 shows the plot of e_1 and e_2 versus time while Figure 11 shows $dist_{1,2}$, i.e., the mutual distance between the two robots, versus time. Finally, Table 2 summarizes results, including the average and standard deviation of $|e_1|$ and $|e_2|$, as well as the minimum distance $md_{1,2} = \min(dist_{1,2})$. This latter value is particularly significant since it is a measure of the safety of the approach, and shows that no collisions have been detected during the experiments since $md_{1,2}$ is always greater than d_{safe}.

The same experiment has been repeated by letting the robots move with different linear speeds $u_1(t)$ (for robot 1) and $u_2(t)$ (for robot 2), keeping A_j and σ constant. The plots of the robot paths are shown in Figure 12, while the error e_1 and e_2 and

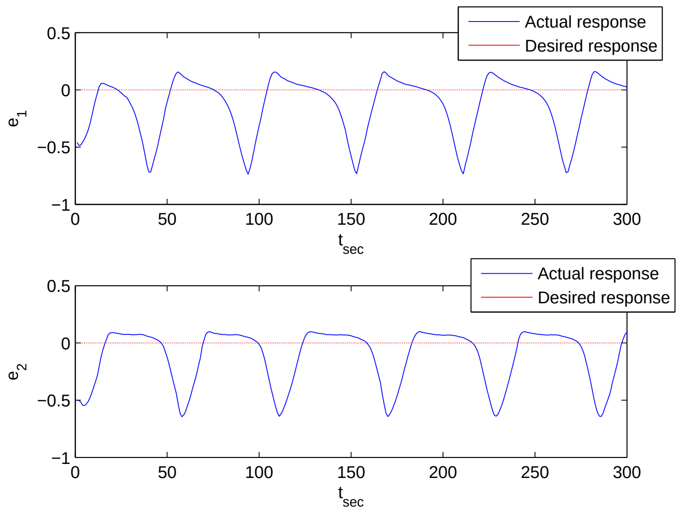

Figure 10: Two Robots with the same speed: Plot of e_1 and e_2 versus time.

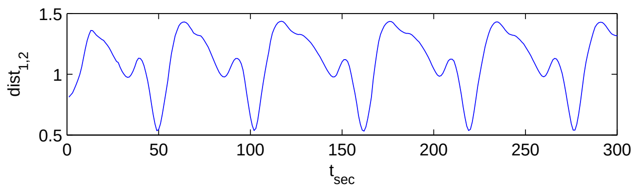

Figure 11: Two Robots with the same speed: Plot of $dist_{1,2}$ versus time.

Table 2: Summary: Response of Two Robots by varying σ with $u_1(t) = u_2(t) = 0.3m/s$.

$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$md_{1,2}$	K_1, K_2	A_j , σ
0.138, 0.097	0.134, 0.116	0.535	15, 2	0.4, 0.4
0.143, 0.089	0.136, 0.099	0.536	15, 2	0.4, 0.45
0.158, 0.121	0.172, 0.135	0.554	15, 2	0.4, 0.5
0.178, 0.128	0.191, 0.135	0.579	15, 2	0.4, 0.55
0.179, 0.111	0.156, 0.083	0.591	15, 2	0.4, 0.6

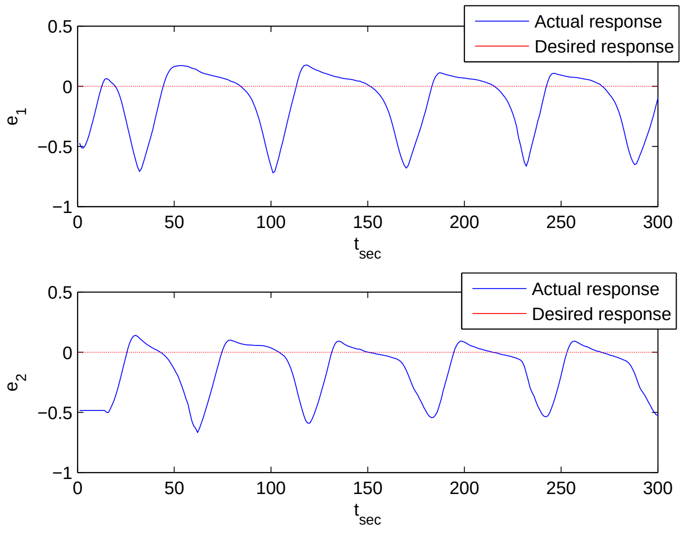

Figure 13: Two Robots with different speed: Plot of e_1 and e_2 versus time.

the distance $dist_{1,4}$ are plotted in Figure 13 and in Figure 14. Table 3 summarizes the results. Notice that, when changing the linear speed, it is also necessary to properly tune the control gains K_1 and K_2, that now turn out to be different for the two robots.

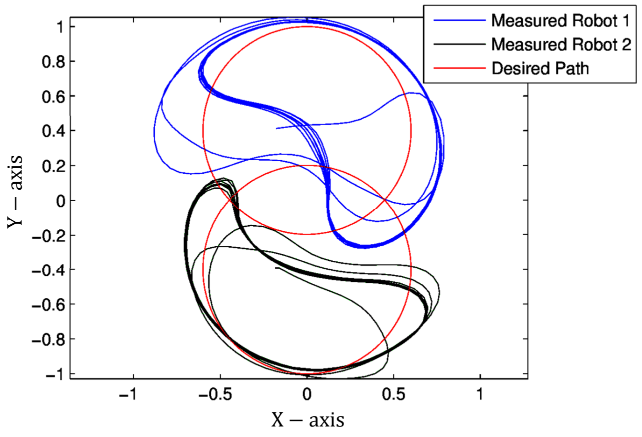

Figure 12: Two Robots with different speed recognizing each other as obstacles.

Robot 1: $u_1(t) = 0.6m/s, K_1 = 10, K_2 = 1, A_j = 0.4, \sigma = 0.55$;

Robot 2: $u_2(t) = 0.5m/s, K_1 = 10, K_2 = 1.2, A_j = 0.4, \sigma = 0.55$.

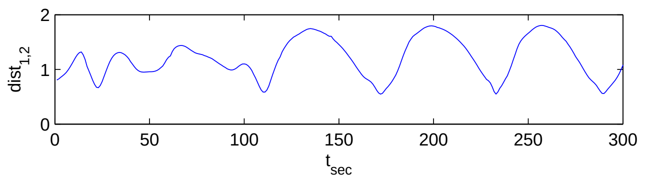

Figure 14: Two Robots with different speed: Plot of $dist_{1,2}$ versus time.

Table 3: Summary: Response of Two Robots by varying speed and $A_j = 0.4$, $\sigma = 0.55$.

$u_1(t)$, $u_2(t)$	$av(\lvert e_1 \rvert)$, $std(\lvert e_1 \rvert)$	$av(\lvert e_2 \rvert)$, $std(\lvert e_2 \rvert)$	$md_{1,2}$	$Robot1$ K_1, K_2	$Robot2$ K_1, K_2
0.3 , 0.1	0.163, 0.134	0.155, 0.136	0.548	15, 2	20, 4
0.4 , 0.5	0.182, 0.122	0.167, 0.112	0.532	12, 1.6	11, 1.2
0.6 , 0.5	0.208, 0.129	0.182, 0.107	0.539	10, 1	11, 1.2

The next test has been performed with a more complex scenario, using three robots. Robots are moving along three intersecting circular paths, setting different values for σ and having different speeds. Figures 15, 16 respectively show the plot of the robot paths and the plot of the mutual distances between robots (i.e., $dist_{1,2}$, $dist_{1,3}$, $dist_{2,3}$). Table 4 and Table 5 summarize the results of the experiments with three robots: Table 4 reports results when having three different linear velocities $u_1(t)$, $u_2(t)$, $u_3(t)$ for the three robots; Table 5 reports results by varying σ.

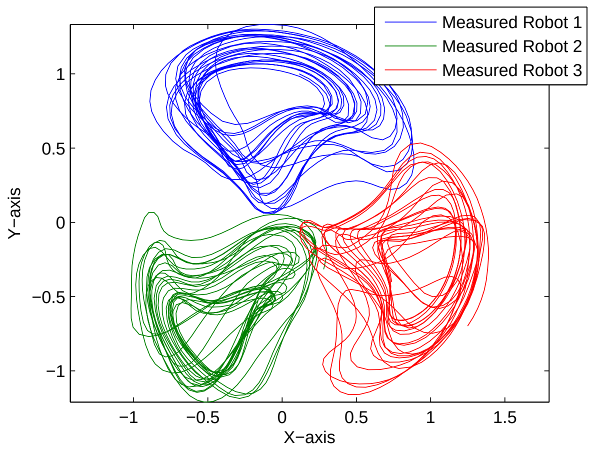

Figure 15: Three Robots with different speed and different σ recognizing each other as obstacles.

Robot 1: $u_1(t) = 0.6m/s, K_1 = 10, K_2 = 1, A_j = 0.45, \sigma = 0.6$;

Robot 2: $u_2(t) = 0.5m/s$, $K_1 = 11$, K_2=1.2 , A_j=0.4, σ=0.55;

Robot 3: $u_3(t) = 0.3m/s, K_1 = 15, K_2 = 2, A_j = 0.6, \sigma = 0.65$.

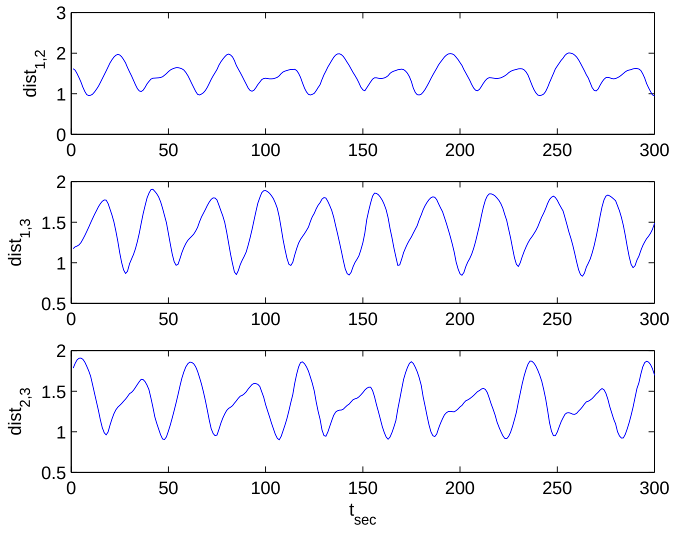

Figure 16: Three Robots with different speed and different σ: Plot of $dist_{1,2}$, $dist_{1,3}$, $dist_{2,3}$ versus time.

In all the Tables in this Subsection, the mutual distance $d_{k,i}$ between agents k and i is always greater than d_{safe} (thus guaranteeing than no collision has occurred) and the average error e_k between the desired and the actual path is always bounded, thus being

Table 4: Summary: Response of Three Robots by varying speed and $A_j = 0.4$, $\sigma = 0.55$.

$u_1(t)$, $u_2(t)$, $u_3(t)$	$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$av(\|e_3\|)$, $std(\|e_3\|)$	$md_{1,2}$	$md_{1,3}$	$md_{2,3}$	$Robot1$ K_1, K_2	$Robot2$ K_1, K_2	$Robot3$ K_1, K_2
0.1, 0.2, 0.3	0.173, 0.119	0.169, 0.138	0.151, 0.130	0.593	0.549	0.677	20, 4	18, 3.5	15, 2
0.3, 0.4, 0.5	0.211, 0.120	0.230, 0.145	0.333, 0.238	0.761	0.564	0.730	15, 2	12, 1.6	11, 1.2
0.6, 0.5, 0.3	0.349, 0.207	0.298, 0.186	0.316, 0.193	0.438	0.492	0.613	10, 1	11, 1.2	15, 2

an index of the fact that robots go back to the desired path after avoiding each others.

Table 5: Summary: Response of Three Robots by varying σ and $u_1(t) = u_2(t) = u_3(t) = 0.3$, K_1=15, K_2=2.

$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$av(\|e_3\|)$, $std(\|e_3\|)$	$md_{1,2}$	$md_{1,3}$	$md_{2,3}$	A_j, σ
0.179, 0.146	0.208, 0.131	0.364, 0.261	0.728	0.686	0.709	0.4, 0.4
0.186, 0.137	0.220, 0.133	0.343, 0.251	0.752	0.698	0.721	0.4, 0.5
0.202, 0.123	0.228, 0.137	0.334, 0.241	0.755	0.699	0.733	0.4, 0.6

4.3.2. Sine wave Path

In this experiment, a sine wave path (8) has been considered as a reference. Figures 17 and 20 show the plot of two robots moving back and forth along a sinusoidal path.

In the first experiment two intersecting paths are considered, which requires the robots (starting from $x_1 = 0, y_1 = 0$ and $x_2 = 0, y_2 = -1.2$) to avoid each other when they are in proximity of the intersection at the same time (similarly to what would happen to two cars approaching a crossroad).

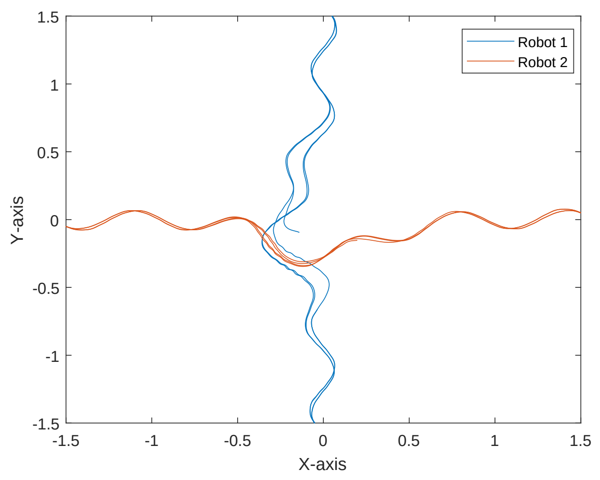

Figure 17: Two Robots moving along two intersecting sine wave paths.

Robot 1: $u_1(t) = 0.3m/s, K_1 = 35, K_2 = 5, A_j = 0.5, \sigma = 0.5$

Robot 2: $u_2(t) = 0.2m/s, K_1 = 40, K_2 = 7, A_j = 0.5, \sigma = 0.5$.

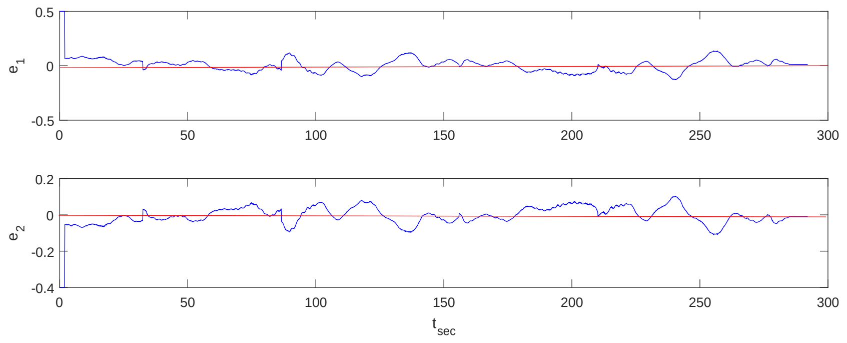

Figure 18: Two Robots moving along two intersecting sine wave paths: Plot of e_1, e_2 versus time.

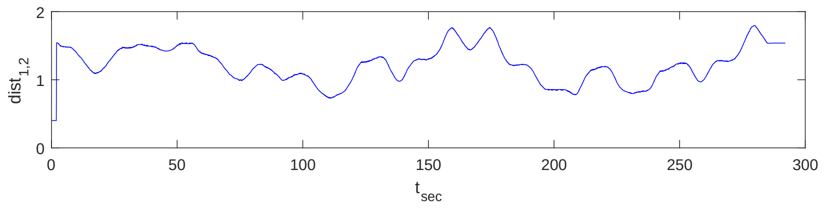

Figure 19: Two Robots moving along two intersecting sine wave paths: Plot of $dist_{1,2}$ versus time.

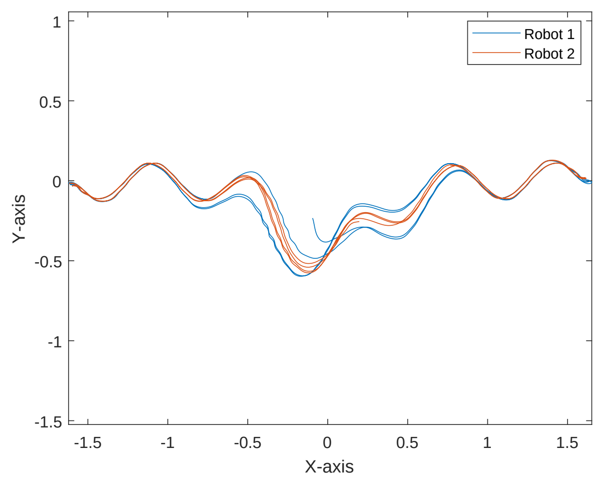

Figure 20: Two Robots moving along the same sine wave path, but in opposite directions.

Robot 1: $u_1(t) = 0.1m/s, K_1 = 45, K_2 = 10, A_j = 0.5, \sigma = 0.5$

Robot 2: $u_2(t) = 0.1m/s, K_1 = 45, K_2 = 10, A_j = 0.5, \sigma = 0.5$.

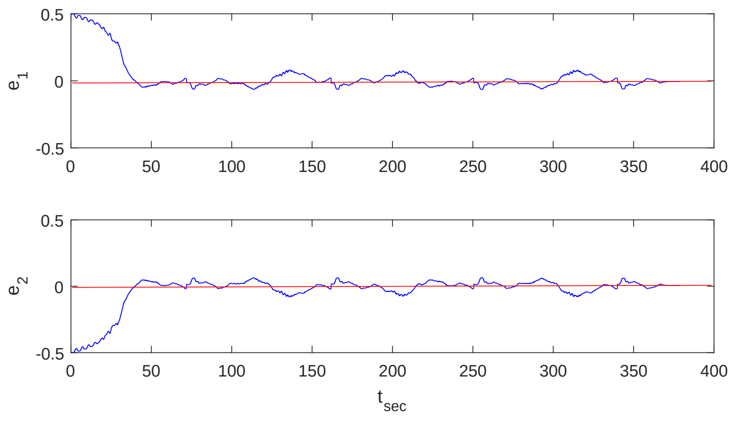

Figure 21: Two Robots moving along the same sine wave path: Plot of e_1, e_2 versus time.

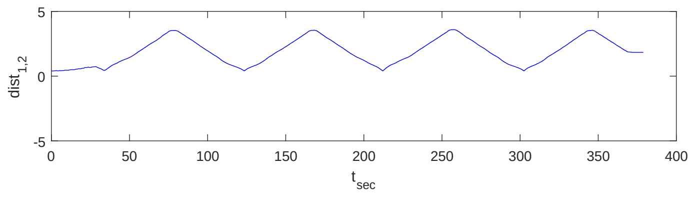

Figure 22: Two Robots moving along the same sine wave path: Plot of $dist_{1,2}$ versus time.

Table 6: Summary: Response of Two Robots moving back and forth along two sine wave paths intersecting each other, by varying speed and $A_j = 0.5$, $\sigma = 0.5$.

u_1, $u_2(t)$	$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$md_{1,2}$	$Robot1$ K_1, K_2	$Robot2$ K_1, K_2
0.1 , 0.1	0.076, 0.069	0.075, 0.071	0.707	45, 10	45, 10
0.2 , 0.1	0.107, 0.099	0.081, 0.073	0.681	40, 7	45, 10
0.3 , 0.2	0.121, 0.117	0.113, 0.109	0.669	35, 5	40, 7

Table 7: Summary: Response of Two Robots moving back and forth along the same sine wave path but in opposite directions, by varying speed and $A_j = 0.5$, $\sigma = 0.5$.

u_1, $u_2(t)$	$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$md_{1,2}$	$Robot1$ K_1, K_2	$Robot2$ K_1, K_2
0.1 , 0.1	0.102, 0.097	0.111, 0.102	0.648	45, 10	45, 10
0.2 , 0.1	0.148, 0.133	0.107, 0.101	0.532	40, 7	45, 10
0.3 , 0.2	0.173, 0.166	0.153, 0.132	0.439	35, 5	40, 7

In the second experiment, the reference path is the same for the two robots, but they move in opposite directions (starting from $x_1 = 0, y_1 = 0$ and $x_2 = 1.5, y_2 = 0$), which requires them to avoid each other whenever they meet somewhere along the path (similarly to what would happen to two cars moving along the same road but in opposite directions). Figures 18, 19, 21, 22, show the errors e_1, e_2 and distance $dist_{1,2}$ between robots while they follow their path back and forth. The control parameters to properly follow the sinusoidal path, along with the results in terms of average error and minimum distance are shown in Table 6 and 7.

4.3.3. Straight Line Path

This experiment is almost identical to the previous one, with the only difference that a straight line (6) has been considered as a reference. In the first case (Figure 23), two intersecting lines are considered (the robots start from $x_1 = 0, y_1 = 0$ and $x_2 = 0, y_2 = -1.2$). In the second case (Figure 26), the two robots move along the same line but in opposite directions (starting from $x_1 = 0, y_1 = -1.2$ and $x_2 = 0, y_2 = 1.2$).

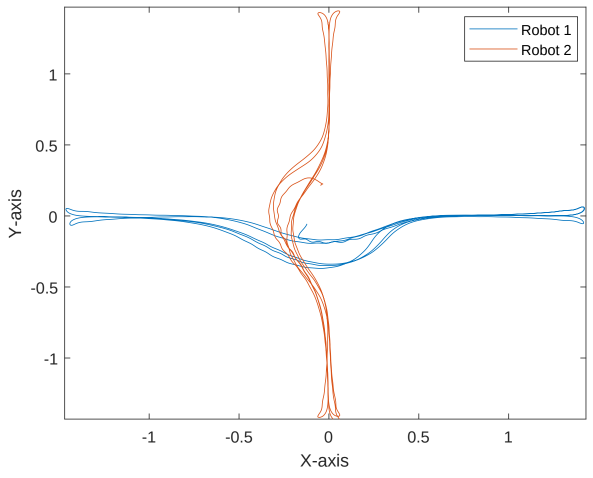

Figure 23: Two Robots moving along two intersecting straight paths.
Robot 1: $u_1(t) = 0.1m/s, K_1 = 25, K_2 = 6, A_j = 0.5, \sigma = 0.5$
Robot 2: $u_2(t) = 0.1m/s, K_1 = 25, K_2 = 6, A_j = 0.5, \sigma = 0.5$.

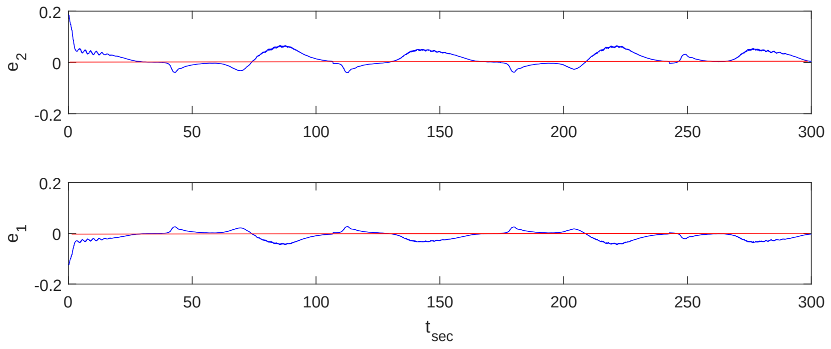

Figure 24: Two Robots moving along two intersecting straight paths: Plot of e_1, e_2 versus time.

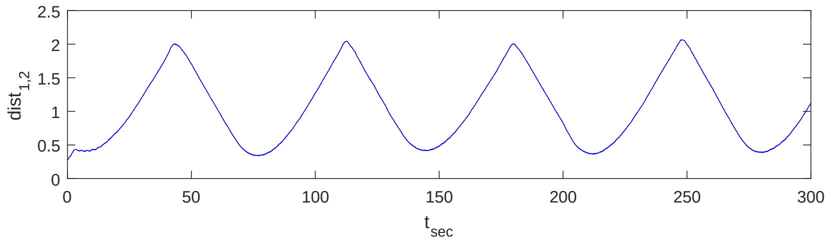

Figure 25: Two Robots moving along two intersecting straight paths: Plot of $dist_{1,2}$ versus time.

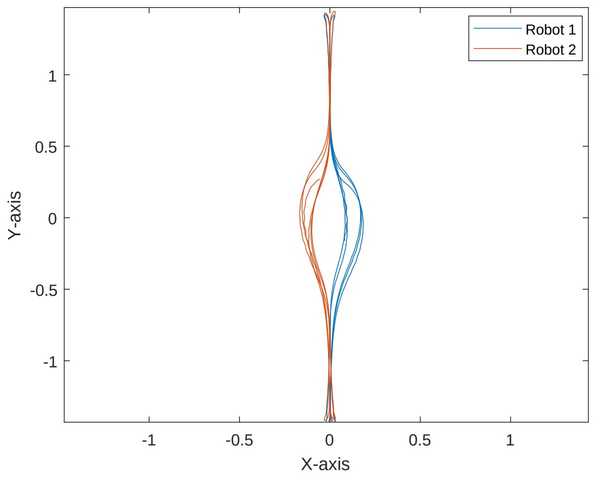

Figure 26: Two Robots moving along the same straight path with the same speed, but in opposite directions.

Robot 1: $u_1(t) = 0.1m/s, K_1 = 25, K_2 = 6, A_j = 0.5, \sigma = 0.5$

Robot 2: $u_2(t) = 0.1m/s, K_1 = 25, K_2 = 6, A_j = 0.5, \sigma = 0.5$.

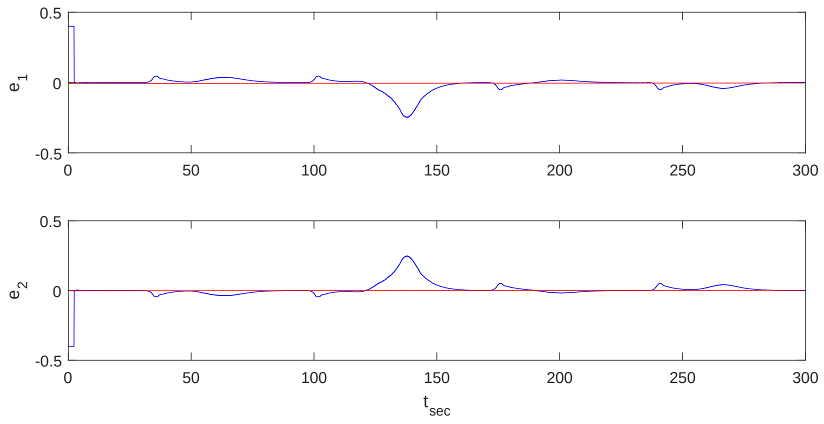

Figure 27: Two Robots moving along the same straight path with the same speed: Plot of e_1, e_2 versus time.

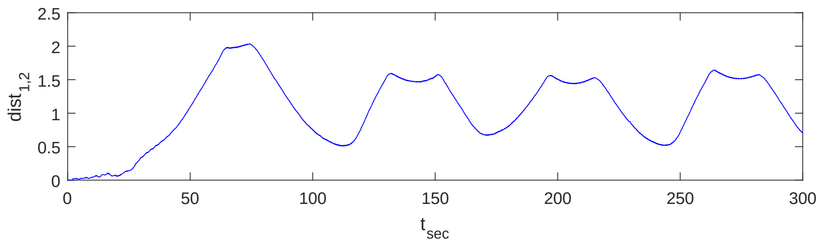

Figure 28: Two Robots moving along the same straight path with the same speed: Plot of $dist_{1,2}$ versus time.

Table 8: Summary: Response of Two Robots moving back and forth along two straight line paths intersecting each other, by varying speed and $A_j = 0.5$, $\sigma = 0.5$.

u_1, $u_2(t)$	$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$md_{1,2}$	$Robot1$ K_1, K_2	$Robot2$ K_1, K_2
0.1 , 0.1	0.097, 0.089	0.089, 0.082	0.567	45, 10	45, 10
0.2 , 0.1	0.125, 0.119	0.131, 0.122	0.452	40, 7	45, 10
0.3 , 0.2	0.170, 0.153	0.159, 0.148	0.439	35, 5	40, 7

Table 9: Summary: Response of Two Robots moving back and forth along the same straight line path but in opposite directions, by varying speed and $A_j = 0.5$, $\sigma = 0.5$.

u_1, $u_2(t)$	$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$md_{1,2}$	$Robot1$ K_1, K_2	$Robot2$ K_1, K_2
0.1 , 0.1	0.086, 0.077	0.079, 0.068	0.492	45, 10	45, 10
0.2 , 0.1	0.122, 0.116	0.092, 0.081	0.337	40, 7	45, 10
0.3 , 0.2	0.127, 0.123	0.119, 0.115	0.266	35, 5	40, 7

Figures 24, 25, 27, 28, shows the errors e_1, e_2 and distance $dist_{1,2}$ between robots while they follow their path back and forth. Figure 29 shows the two robots moving along the same line, but now with a different speed (starting from $x_1 = -1.2, y_1 = 0$ and $x_2 = 1.2, y_2 = 0$) and its error and distance shown in Figures 30, 31. The control parameters to properly follow the straight line path, along with the results in terms of average error and minimum distance are shown in Tables 8 and 9.

4.4. Scenario 4: With Persons

In this scenario, one or multiple robots move along a circular path at constant speed in the motion capture area, but this time concurrently with persons walking in the same area: then, the robots are obstructed in following the path by the presence of walking persons, as shown in Figure 32. While the persons walk in a random way, the robots have been given a speed of 0.3m/s, and the gains are set according to the analysis performed in the previous experiments. Indeed, persons have been considered in the experiments because their motion is less repeatable and predictable with respect to

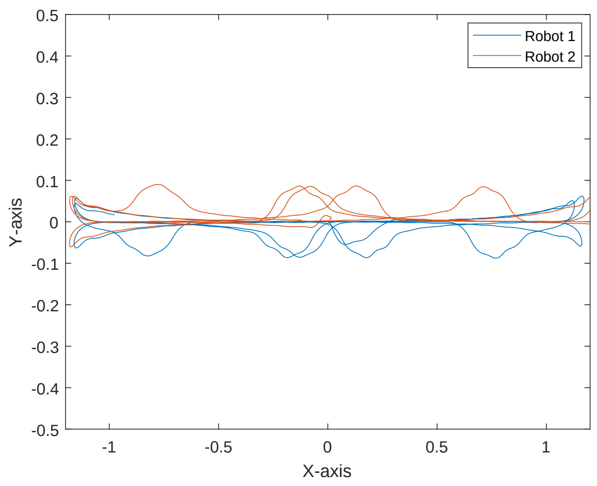

Figure 29: Two Robots moving along the same straight line path in opposite directions, but with different speed.

Robot 1: $u_1(t) = 0.3m/s, K_1 = 35, K_2 = 5, A_j = 0.5, \sigma = 0.5$

Robot 2: $u_2(t) = 0.2m/s, K_1 = 40, K_2 = 7, A_j = 0.5, \sigma = 0.5.$

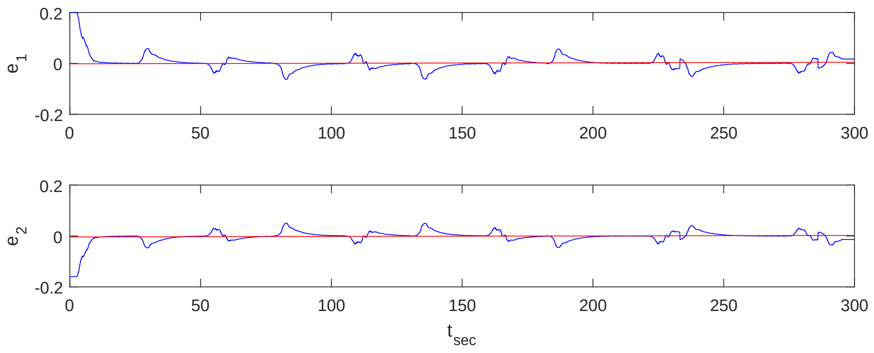

Figure 30: Two robots moving along the same straight line path but with different speed: Plot of $e_{1,2}$ versus time.

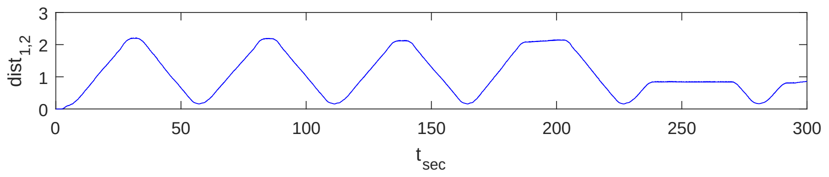

Figure 31: Two robots moving along the same straight line path but with different speed: Plot of $dist_{1,2}$ versus time.

Figure 32: Robots with Persons.

mobile robots, thus introducing an element of variability in the robot's path.

In the first case study, the response of one robot with one walking person is considered. The actual path of the robot in presence of one person is shown in Figure 33. The path followed by the robot is in this case much more subject to disturbances, because the random interference of the walking person tends to be more frequent with respect to the multi-robot case, Figure 34. The distance between the robot and the person is shown in Figure 35.

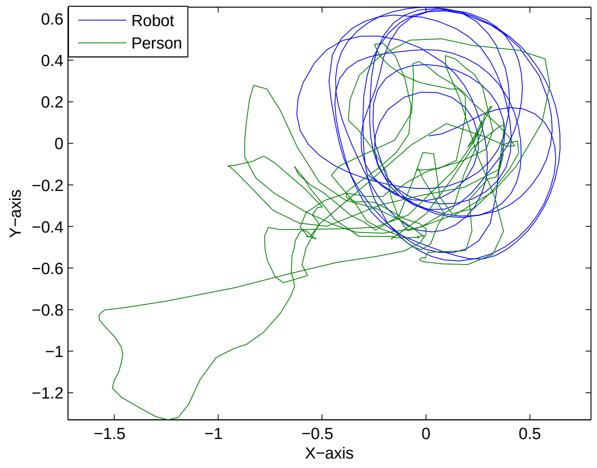

Figure 33: One Robot with one Person: $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$.

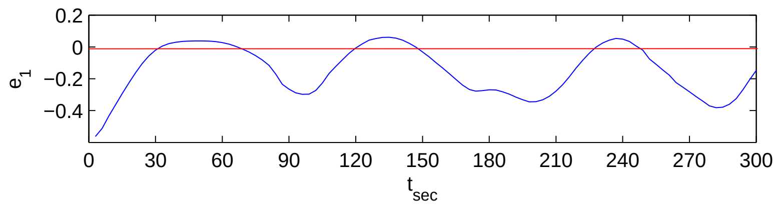

Figure 34: One Robot with one Person: Plot of e_1 versus time.

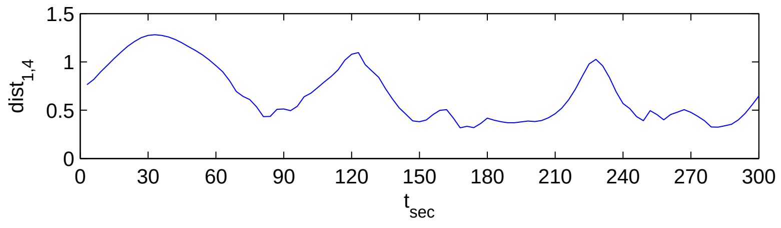

Figure 35: One Robot with one Person: Plot of $dist_{1,4}$ versus time.

In the next case study, a robot and two persons are considered. The path followed by the robot and the two persons is shown in Figure 36.

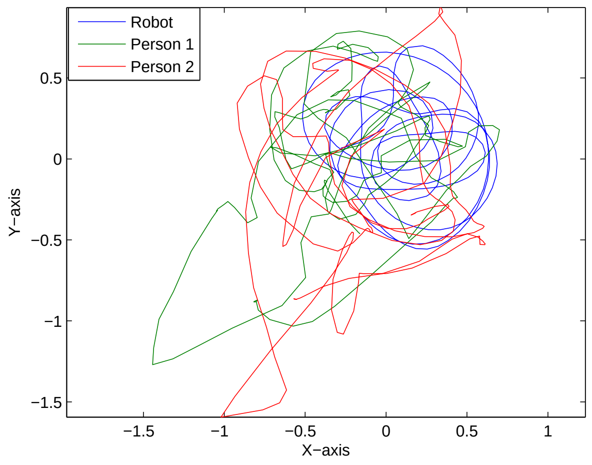

Figure 36: One Robot with Two Persons: $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$.

The same experiment has been performed 3 times for each case study (1, 2 and 3 persons). All results are presented in Table 10. The analysis of the values of $av(|e_1|)$

Table 10: Summary: Response of One Robot with Persons, $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$

No. Persons	$av(\|e_1\|)$, $std(\|e_1\|)$	$md_{1,4}$	$md_{1,5}$	$md_{1,6}$
1	0.092, 0.067	0.457	—	—
2	0.142, 0.101	0.437	0.418	—
3	0.266, 0.198	0.380	0.551	0.274

Table 11: Summary: Response of Two Robots with Persons, $u_1(t) = u_2(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$.

No. Persons	$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$md_{1,2}$	$md_{1,4}$, $md_{2,4}$	$md_{1,5}$, $md_{2,5}$	$md_{1,6}$, $md_{2,6}$
1	0.160, 0.119	0.219, 0.137	0.397	0.452, 0.413	—	—
2	0.181, 0.124	0.242, 0.155	0.297	0.391, 0.431	0.417, 0.521	—
3	0.211, 0.156	0.294, 0.196	0.503	0.427, 0.419	0.321, 0.403	0.381, 0.410

and $std(|e_1|)$ shows that the robot diverges from its path while persons are crossing it. The minimum distance of the robot from each person $md_{1,i}$ is reported in the table.

Similar tests have been performed by adding more robots (totally 2 and 3) moving along predetermined circular paths with 1, 2 and 3 persons walking randomly in the same area. Figures 37, 38 and Tables 11, 12 show the obtained results. As usual, $dist_{k,i}$ is the distance between robot k and robot or persons i, while $av(|e_k|)$ and $std(|e_k|)$ are the average error and the standard deviation between the predefined path and the actual path of robot k.

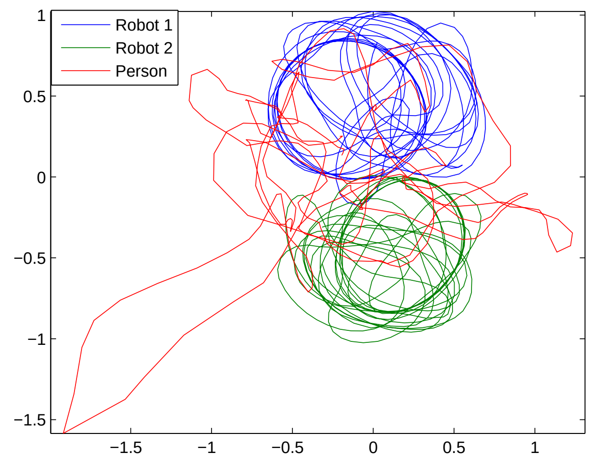

Figure 37: Two Robots with One Person:

Robot 1: $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$

Robot 2: $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$.

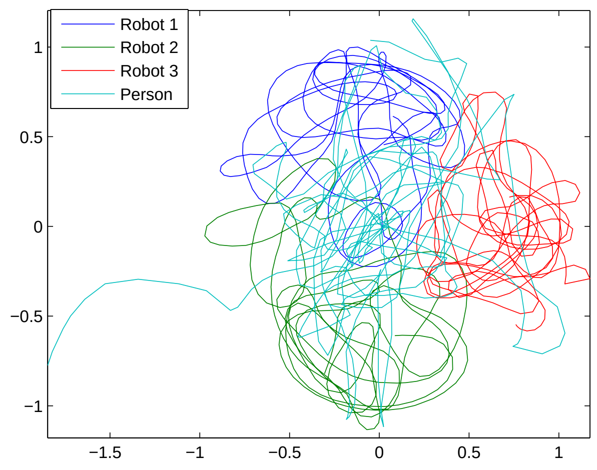

Figure 38: Three Robots with One Person:

Robot 1: $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$

Robot 2: $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$

Robot 3: $u_1(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2$, $A_j = 0.4$, $\sigma = 0.45$.

Table 12: Summary: Response of Three Robots with Persons, $u_1(t) = u_2(t) = u_3(t) = 0.3m/s$, $K_1 = 15$, $K_2 = 2, A_j = 0.4$, $\sigma = 0.45$.

No. Persons	$av(\|e_1\|)$, $std(\|e_1\|)$	$av(\|e_2\|)$, $std(\|e_2\|)$	$av(\|e_3\|)$, $std(\|e_3\|)$	$md_{1,2}$, $md_{1,3}$, $md_{2,3}$	$md_{1,4}$, $md_{2,4}$, $md_{3,4}$	$md_{1,5}$, $md_{2,5}$, $md_{3,5}$	$md_{1,6}$, $md_{2,6}$, $md_{3,6}$
1	0.166, 0.104	0.336, 0.215	0.121, 0.087	0.595, 0.521, 0.505	0.404, 0.391, 0.457	—	—
2	0.237, 0.153	0.214, 0.149	0.209, 0.177	0.472, 0.495, 0.517	0.532, 0.438, 0.427	0.507, 0.465, 0.323	—
3	0.118, 0.072	0.298, 0.116	0.411, 0.192	0.432, 0.497, 0.322	0.288, 0.216, 0.411	0.439, 0.337, 0.355	0.468, 0.506, 0.444

5. Conclusions and Discussion

In this article an integrated approach for path following and obstacle avoidance has been discussed, while considering persons and robots as moving obstacles. The proposed approach has been validated by quantitatively measuring its performance in a $3m \times 3m$ meters arena crowded with robots and persons. The experimental results, obtained with up to three mobile wheeled robots, confirm the robustness and the safety of the approach even in complex scenarios with moving obstacles and persons, in very narrow areas and at significant velocities.

As anticipated in Section 1, the reader may find some similarities with Artificial Potential Fields and other force field-based methods [20][19]. In particular, this resemblance is a consequence of the fact that, similarly to force field-based methods, the proposed approach reactively adds a contribution for each locally-sensed obstacle. However, the proposed approach is different for two main reasons, which follows the fact that both the initial path as well as the deformed path are described as curves expressed through their implicit equations, respectively $f(x,y) = 0$ and $f'(x,y) = 0$.

- Differently from force field-based methods, the proposed approach guarantees that there the robot can never be stuck in a position x, y without a preferred direction to move (in force field-based methods, this happens in correspondence of local minima, which are very frequent in presence of multiple obstacles). This property of the approach is due to the fact that the function $f(x,y)$ as well as the obstacle functions $O_j(x,y)$ are twice differentiable functions in $\mathscr{R}^2$, and therefore the deformed path $f'(x,y) = 0$ is necessarily continuous in $\mathscr{R}^2$, i.e., a direction to proceed along the path is always uniquely defined (see Figure 3).

- The error from the deformed path can be computed by simply evaluating $f'(x,y)$ in the robot's position x, y, and then fed the result to a feedback controller [16] which guarantees asymptotic convergence to the path. This ultimately allows for setting the control variables (linear and angular speed) as a unique continuous function of the deformed path $f'(x,y) = 0$, the robot's pose x, y, θ, and the relative position x_j, y_j of all the locally-sensed obstacles with respect to the robot.

From the analysis of the experiments some general conclusions can be drawn:

- The parameter K_1 and K_2 should be selected properly depending on the desired path and speed.

- The parameters σ and A_j of the Gaussian (obstacle) function should be opportunely tuned in order to avoid the obstacles. A procedure has been introduced that makes possible to set the two vaues in such a way to guarantee that no collision can be produced. By arbitrarily increasing σ and A_j, the distance between the robot and the obstacles increases, but the average error between the actual path and the desired path increases as well.

- The approach proves to work correctly in static environments (*Scenarios* $1, 2$), since the error between the robots and its path is always less than 0.07m with no obstacle and less than 0.24m with static obstacle, as shown in Table 1.

- While multiple robots are moving together (*Scenario* 3), varying A and σ has the effect of avoiding obstacle. As robot speed increases, the appropriate gain tuning parameters must be selected in order to stabilize the robot response. Results are shown in Tables 2, 3, 4, 5, 7, 6, 8 and 9.

- In the case of multiple robots and multiple walking persons (*Scenario* 4) increasing the number of robots and persons will increase the path following error ($av|e|$), while decreasing the average distance between persons and robots $av(|dist_{min}|)$. This is shown in Tables 10, 11 and 12.
- In scenario 4, the tracking error increases, but the performance can be still considered satisfactory. On the other hand, the proposed method shows significantly performance with static obstacles.

Further tests are planned for the next future in real scenarios with more complex paths and a higher number of moving obstacles, and by considering different sources of information (i.e., not MoCap-based) to compute the robot's position.

Acknowledgement

The work has been supported by MIUR (Italian Ministry of Education, Universities and Research) through the project MAREA (Monitoring And Rescue Automation) coordinated by SIIT (Technological District of Liguria on Integrated Intelligent Systems).

6. References

[1] A. S. Matveev, H. Teimoori, A. V. Savkin, A method for guidance and control of an autonomous vehicle in problems of border patrolling and obstacle avoidance, Automatica 47 (3) (2011) 515–524.

[2] F. Bonin-Font, A. Ortiz, G. Oliver, Visual navigation for mobile robots: A survey, Journal of intelligent and robotic systems 53 (3) (2008) 263.

[3] A. R. Mosteo, E. Montijano, D. Tardioli, Optimal role and position assignment in multi-robot freely reachable formations, Automatica 81 (2017) 305–313.

[4] D. W. Oyler, P. T. Kabamba, A. R. Girard, Pursuit–evasion games in the presence of obstacles, Automatica 65 (2016) 1–11.

[5] A. Doosthoseini, C. Nielsen, Coordinated path following for unicycles: A nested invariant sets approach, Automatica 60 (2015) 17–29.

[6] Y. Kantaros, M. M. Zavlanos, Distributed communication-aware coverage control by mobile sensor networks, Automatica 63 (2016) 209–220.

[7] X. Li, H. Yang, J. Wang, D. Sun, Design of a robust unified controller for cell manipulation with a robot-aided optical tweezers system, Automatica 55 (2015) 279–286.

[8] X. Li, D. Sun, J. Yang, A bounded controller for multirobot navigation while maintaining network connectivity in the presence of obstacles, Automatica 49 (1) (2013) 285–292.

[9] A. S. Matveev, M. Hoy, A. V. Savkin, A method for reactive navigation of non-holonomic under-actuated robots in maze-like environments, Automatica 49 (5) (2013) 1268–1274.

[10] A. V. Savkin, C. Wang, Seeking a path through the crowd: Robot navigation in unknown dynamic environments with moving obstacles based on an integrated environment representation, Robotics and Autonomous Systems 62 (10) (2014) 1568–1580.

[11] L. Xiao-Qing, W. Yao-Nan, M. Jian-Xu, Nonlinear control for multi-agent formations with delays in noisy environments, Acta Automatica Sinica 40 (12) (2014) 2959–2967.

[12] Y. Su, Leader-following rendezvous with connectivity preservation and disturbance rejection via internal model approach, Automatica 57 (2015) 203–212.

[13] X. Sun, C. G. Cassandras, Optimal dynamic formation control of multi-agent systems in constrained environments, Automatica 73 (2016) 169–179.

[14] Y. Tian-Tian, L. Zhi-Yuan, C. Hong, P. Run, Formation control and obstacle avoidance for multiple mobile robots, Acta Automatica Sinica 34 (5) (2008) 588–593.

[15] M. Malisoff, R. Sizemore, F. Zhang, Adaptive planar curve tracking control and robustness analysis under state constraints and unknown curvature, Automatica 75 (2017) 133–143.

[16] A. Morro, A. Sgorbissa, R. Zaccaria, Path following for unicycle robots with an arbitrary path curvature, IEEE Transactions on Robotics 27 (5) (2011) 1016–1023.

[17] M. Michalek, A highly scalable path-following controller for n-trailers with off-axle hitching, Control Engineering Practice 29 (2014) 61–73, cited By 5.

[18] A. Sgorbissa, R. Zaccaria, 3d path following with no bounds on the path curvature through surface intersection, in: Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Taipei, Taiwan, 2010.

[19] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, The International Journal of Robotics Research 5 (1) (1986) 90–98.

[20] J. Borenstein, Y. Koren, The vector field histogram-fast obstacle avoidance for mobile robots, IEEE Transactions on Robotics and Automation 7 (3) (1991) 278 –288.

[21] A. V. Savkin, C. Wang, A simple biologically inspired algorithm for collision-free navigation of a unicycle-like robot in dynamic environments with moving obstacles, Robotica 31 (06) (2013) 993–1001.

[22] P. D. H. Nguyen, C. T. Recchiuto, A. Sgorbissa, Real-time path generation and obstacle avoidance for multirotors: A novel approach, Journal of Intelligent & Robotic Systems 89 (1) (2018) 27–49.

[23] C. Possieri, A. R. Teel, Lq optimal control for a class of hybrid systems, in: Decision and Control (CDC), 2016 IEEE 55th Conference on, IEEE, 2016, pp. 604–609.

[24] M. Hoy, A. S. Matveev, A. V. Savkin, Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey, Robotica 33 (03) (2015) 463–497.

[25] J. Minguez, F. Lamiraux, J.-P. Laumond, Motion planning and obstacle avoidance, in: Springer handbook of robotics, Springer, 2016, pp. 1177–1202.

[26] A. V. Savkin, A. S. Matveev, M. Hoy, C. Wang, Safe robot navigation among moving and steady obstacles, Butterworth-Heinemann, 2015.

[27] E. Garone, S. Di Cairano, I. Kolmanovsky, et al., Reference and command governors for systems with constraints: A survey on theory and applications, Automatica 75 (2017) 306–328.

[28] G. M. Atınç, D. M. Stipanović, P. G. Voulgaris, Supervised coverage control of multi-agent systems, Automatica 50 (11) (2014) 2936–2942.

[29] K. Zhang, J. Sprinkle, R. G. Sanfelice, A hybrid model predictive controller for path planning and path following, in: Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems, ACM, 2015, pp. 139–148.

[30] S. Brüggemann, C. Possieri, J. I. Poveda, A. R. Teel, Robust constrained model predictive control with persistent model adaptation, in: Decision and Control (CDC), 2016 IEEE 55th Conference on, IEEE, 2016, pp. 2364–2369.

[31] L. Fagiano, A. R. Teel, Model predictive control with generalized terminal state constraint, IFAC Proceedings Volumes 45 (17) (2012) 299–304.

[32] X. Yu-Geng, L. De-Wei, L. Shu, Model predictive controlstatus and challenges, Acta Automatica Sinica 39 (3) (2013) 222–236.

[33] D. Liberzon, Calculus of variations and optimal control theory: a concise introduction, Princeton University Press, 2012.

[34] G. Pin, D. M. Raimondo, L. Magni, T. Parisini, Robust model predictive control of nonlinear systems with bounded and state-dependent uncertainties, IEEE Transactions on automatic control 54 (7) (2009) 1681–1687.

[35] G. C. Goodwin, R. H. Middleton, M. M. Seron, B. Campos, Application of nonlinear model predictive control to an industrial induction heating furnace, Annual Reviews in Control 37 (2) (2013) 271–277.

[36] M. S. Rana, H. R. Pota, I. R. Petersen, The design of model predictive control for an afm and its impact on piezo nonlinearities, European Journal of Control 20 (4) (2014) 188–198.

[37] D. Kouzoupis, A. Zanelli, H. Peyrl, H. Ferreau, Towards proper assessment of qp algorithms for embedded model predictive control, in: Control Conference (ECC), 2015 European, IEEE, 2015, pp. 2609–2616.

[38] W. Kim, D. Kim, K. Yi, H. J. Kim, Development of a path-tracking control system based on model predictive control using infrastructure sensors, Vehicle System Dynamics 50 (6) (2012) 1001–1023.

[39] R. V. Parys, G. Pipeleers, Distributed MPC for multi-vehicle systems moving in formation, Robotics and Autonomous Systems 97 (2017) 144–152. `doi:10.1016/j.robot.2017.08.009`.
URL `https://doi.org/10.1016/j.robot.2017.08.009`

[40] G. Franzè, W. Lucia, An obstacle avoidance model predictive control scheme for mobile robots subject to nonholonomic constraints: A sum-of-squares approach, Journal of the Franklin Institute 352 (6) (2015) 2358–2380.

[41] T. P. Nascimento, A. G. Conceicao, A. P. Moreira, Multi-robot systems formation control with obstacle avoidance, IFAC Proceedings Volumes 47 (3) (2014) 5703–5708. `doi:10.3182/20140824-6-za-1003.01848`.
URL `https://doi.org/10.3182/20140824-6-za-1003.01848`

[42] S. Bououden, M. Chadli, H. R. Karimi, An ant colony optimization-based fuzzy predictive control approach for nonlinear processes, Information Sciences 299 (2015) 143–158.

[43] P. Fiorini, Z. Shiller, Motion planning in dynamic environments using the relative velocity paradigm, in: Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on, IEEE, 1993, pp. 560–565.

[44] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, T. L. Huntsberger, Safe maritime autonomous navigation with colregs, using velocity obstacles, IEEE Journal of Oceanic Engineering 39 (1) (2014) 110–119.

[45] F. Large, C. Laugier, Z. Shiller, Navigation among moving obstacles using the nlvo: Principles and applications to intelligent vehicles, Autonomous Robots 19 (2) (2005) 159–171.

[46] A. V. Savkin, C. Wang, A simple biologically inspired algorithm for collision-free navigation of a unicycle-like robot in dynamic environments with moving obstacles, Robotica 31 (06) (2013) 993–1001. `doi:10.1017/s0263574713000313`.
URL `https://doi.org/10.1017/s0263574713000313`

[47] O. Montiel, U. Orozco-Rosas, R. Sepúlveda, Path planning for mobile robots using bacterial potential field for avoiding static and dynamic obstacles, Expert Systems with Applications 42 (12) (2015) 5177–5191.

[48] M. Mujahed, D. Fischer, B. Mertsching, Admissible gap navigation: A new collision avoidance approach, Robotics and Autonomous Systems 103 (2018) 93–110. `doi:10.1016/j.robot.2018.02.008`.
URL `https://doi.org/10.1016/j.robot.2018.02.008`

[49] Z. Xu, R. Hess, K. Schilling, Constraints of potential field for obstacle avoidance on car-like mobile robots, IFAC Proceedings Volumes 45 (4) (2012) 169–175. `doi:10.3182/20120403-3-de-3010.00077`.
URL `https://doi.org/10.3182/20120403-3-de-3010.00077`

[50] M. Korayem, S. Nekoo, The SDRE control of mobile base cooperative manipulators: Collision free path planning and moving obstacle avoidance, Robotics and Autonomous Systems 86 (2016) 86–105. `doi:10.1016/j.robot.2016.09.003`.
URL `https://doi.org/10.1016/j.robot.2016.09.003`

[51] A. Sgorbissa, Integrated robot planning, obstacle avoidance, and path following in 2d and 3d: ground, aerial, and underwater vehicles (2017) -`doi:10.13140/rg.2.2.14838.80969`.

[52] R. Al-Jarrah, M. Al-Jarrah, H. Roth, A novel edge detection algorithm for mobile robot path planning, Journal of Robotics 2018.

[53] R. Deepu, B. Honnaraju, S. Murali, Path generation for robot navigation using a single camera, Procedia Computer Science 46 (2015) 1425–1432.

[54] R. Falconi, L. Sabattini, C. Secchi, C. Fantuzzi, C. Melchiorri, Edge-weighted consensus-based formation control strategy with collision avoidance, Robotica 33 (2) (2015) 332–347.

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

[56] K. O. Arras, J. Persson, N. Tomatis, R. Siegwart, Real-time obstacle avoidance for polygonal robots with a reduced dynamic window, in: Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on, Vol. 3, IEEE, 2002, pp. 3050–3055.

[57] P. Ogren, N. E. Leonard, A convergent dynamic window approach to obstacle avoidance, IEEE Transactions on Robotics 21 (2) (2005) 188–195.

[58] B. Damas, J. Santos-Victor, Avoiding moving obstacles: the forbidden velocity map, in: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, IEEE, 2009, pp. 4393–4398.

[59] L. Lapierre, R. Zapata, P. Lepinay, Simulatneous path following and obstacle avoidance control of a unicycle-type robot, in: Robotics and Automation, 2007 IEEE International Conference on, IEEE, 2007, pp. 2617–2622.

[60] M. Aicardi, G. Casalino, A. Bicchi, A. Balestrino, Closed loop steering of unicycle like vehicles via lyapunov techniques, IEEE Robotics & Automation Magazine 2 (1) (1995) 27–35.

[61] D. Soetanto, L. Lapierre, A. Pascoal, Adaptive, non-singular path-following control of dynamic wheeled robots, in: Decision and Control, 2003. Proceedings. 42nd IEEE Conference on, Vol. 2, IEEE, 2003, pp. 1765–1770.

[62] H. Miao, Y.-C. Tian, Dynamic robot path planning using an enhanced simulated annealing approach, Applied Mathematics and Computation 222 (2013) 420–437. `doi:10.1016/j.amc.2013.07.022`.
URL `https://doi.org/10.1016/j.amc.2013.07.022`

[63] A. Mohammadi, M. Rahimi, A. A. Suratgar, A new path planning and obstacle avoidance algorithm in dynamic environment, in: 2014 22nd Iranian Conference on Electrical Engineering (ICEE), IEEE, 2014.

[64] T. Mylvaganam, M. Sassano, Autonomous collision avoidance for wheeled mobile robots using a differential game approach, European Journal of Control 40 (2018) 53–61. `doi:10.1016/j.ejcon.2017.11.005`.
URL `https://doi.org/10.1016/j.ejcon.2017.11.005`

[65] L. Hsien-I, 2d-span resampling of bi-RRT in dynamic path planning, International Journal of Automation and Smart Technology 4 (4) (2015) 39–48. `doi:10.5875/ausmt.v5i1.837`.
URL `https://doi.org/10.5875/ausmt.v5i1.837`

[66] C.-b. Moon, W. Chung, Kinodynamic planner dual-tree rrt (dt-rrt) for two-wheeled mobile robots using the rapidly exploring random tree, IEEE Transactions on industrial electronics 62 (2) (2015) 1080–1090.

[67] D. Chwa, Robust distance-based tracking control of wheeled mobile robots using vision sensors in the presence of kinematic disturbances, IEEE Transactions on Industrial Electronics 63 (10) (2016) 6172–6183. `doi:10.1109/TIE.2016.2590378`.