Multi-sensor Navigation for Soccer Robots

Carlos F. Marques and Pedro U. Lima

Instituto de Sistemas e Robótica
Instituto Superior Técnico, Av. Rovisco Pais, 1 — 1049-001 Lisboa, PORTUGAL
{cmarques,pal}@isr.ist.utl.pt
http://socrob.isr.ist.utl.pt/

Abstract. *This work introduces a method for robot navigation in structured indoors environments, based on the information of multiple sensors. Guidance control is based on odometry, reset at some time instants by a vision-based self-localization algorithm introduced in previous work. Sonar data is used to avoid and go around obstacles. Results from the application of the complete navigation system to a real robot moving on a RoboCup soccer field are presented.*

1 Introduction

Navigation is the robot subsystem that allows the robot to determine its current posture $\xi_i=(x_i, y_i, \theta_i)$ and move autonomously from an initial posture ξ_i to another target posture $\xi_f=(x_f, y_f, \theta_f)$, without colliding with obstacles. Figure 1.a) shows a block diagram of a Navigation system. The robot vehicle can be characterized by its Dynamics and Kinematics. The wheel angular velocities are controlled in closed loop, by the closed loop actuator controllers. The Guidance controller adjusts the vehicle trajectory, by generating references for the closed loop actuator controllers, so as to take the robot from the initial posture to the target posture, avoiding the obstacles in between. The Self-localization algorithm plus the odometry system estimate the robot posture.

The ability to navigate at relatively high speeds through an environment cluttered with static and dynamic obstacles is a crucial issue for a mobile robot. Most robotic tasks require a robot to move to target postures adequate to carry out its planned activities. In robotic soccer, this includes facing the opponent goal with the ball in between or covering the team goal by positioning itself between the ball and the goal, while avoiding the field walls and the other (stopped and moving) robots.

The most usual approach to mobile robot navigation consists of planning a path between the current and the target postures [1,10]. Path planning is typically based on a priori or sensor-based full knowledge of the surrounding environment. Furthermore, due to the robot kinematic constraints, the planned path must be converted into feasible motion. The whole procedure is thus time-consuming, especially when frequent re-planning is needed to handle dynamic environments [1]. Therefore, this is not the most convenient method for high speed motion within environments cluttered with dynamic obstacles. An elegant

A. Birk, S. Coradeschi, and S. Tadokoro (Eds.): RoboCup 2001, LNAI 2377, pp. 144–153, 2002.

solution consists of formulating the guidance problem for non-holonomic vehicles as one of reaching a desired final posture through time-varying non-linear state feedback [9]. This method allows on-line changes to the desired final posture, and ensures stabilization in the final posture, but the incorporation of (even static) obstacles is not considered. It may also lead to large and/or oscillating trajectories. The Potential Field Method [4,7,8] attempts to handle these problems by adding a repulsive acceleration originated by sensed obstacles and an attractive acceleration vector that pulls the robot towards the target posture. Unfortunately, in its original formulation, this method has some shortcomings, such as the local minima problem, when the robot gets inside an U-shaped obstacle, oscillations in the presence of obstacles, no passage between closely spaced obstacles and oscillations in narrow passages [2]. An alternative is the Vector Field Histogram (VFH) [3], method, where the robot finds the open spaces between the obstacles and chooses the heading closest to the target as the direction for safe motion. Minguez and Montano developed the Nearness Diagram Navigation (NDN) method [6] where the VFH is used, but in this case two polar diagrams are introduced: the Nearness Diagram from the Central Point (PND), to find the safe passages in between the obstacles, called valleys, and the Nearness Diagram from the Robot (RND), used to verify the robot safety conditions. The work described in [2,6] is applied to holonomic robots only.

In this paper we concentrate on guidance control and introduce a guidance control method for non-holonomic (differential drive) vehicles, based on odometry reset by a vision-based self-localization algorithm described in a previous paper [5], endowed with sonar-based obstacle avoidance. The guidance controller is used in the field robots of the RoboCup middle-size league ISocRob team, fully integrated in the state machine that coordinates task execution. The odometry is reset at specific states, such as at restart time or before returning to home position. The algorithm can be generally applied to structured indoors environments, provided that visual features can be observed by the self-localization method [5].

2 The Freezone Method

Borenstein et al.[3], introduced the concept of Vector Field Histogram (VFH), based on a two-stage data reduction technique, with three levels of data representation. The first level is the description of the robot environment, where a two-dimensional Cartesian histogram grid is continuously updated. In the second level the Cartesian histogram is mapped onto an one-dimensional Polar Histogram that describes the density of obstacles around the robot. In the last level, the method determines the open spaces among the obstacles (valleys), and uses them as candidates to possible robot trajectories. The method selects the valley closer to the direction to the target.

The Nearness Diagram Navigation, introduced by Minguez and Montano[6], can be described as follows. First, two nearness diagrams are introduced: the Nearness Diagram from the Central Point (PND), used to find the valleys and one

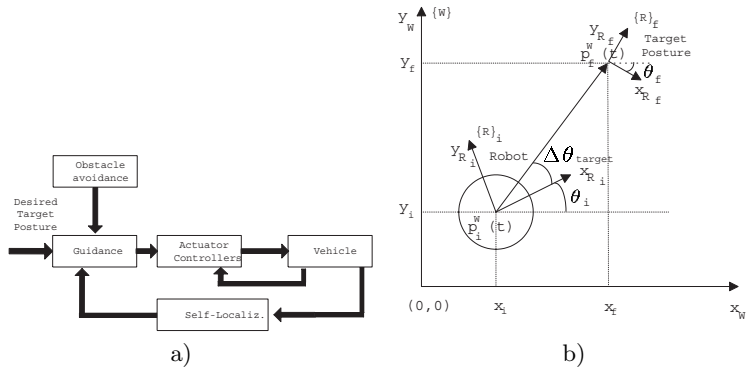


Fig. 1. a) Navigation System. b) Coordinate Systems .

of the regions is selected for the robot to pass, and the Nearness Diagram from the Robot (RND), used to evaluate the robot safety situation. After selecting the free region, and analyzing the safety situation of the robot, the algorithm chooses one navigation strategy based on 5 different heuristics that cover all safety cases. The navigation heuristic chooses the steering angle for the safe path of the robot and, based on this angle, determines the rotational and translational velocities for the robot.

We have developed a guidance control method, named Freezone, that merges the VFH and NDN algorithms, adapting them for non-holonomic (differential drive) robots. Some modifications were introduced, namely the use of a spatial mask that allows finding the valley candidates, and the introduction of a heuristic that allows to avoid oscillations when a small obstacle is in between the robot and the target position.

The coordinate systems used in this work are presented in Fig. 1.b). In the figure, $p_i^W(t) = (x_i^W(t), y_i^W(t))$, is the position of the robot center in the world frame $\{W\}$ at each time instant, $\theta_i(t)$ is its heading in the same frame and $\Delta\theta_{target}$ is the angular error of the robot heading at the initial posture with respect to (w.r.t) the heading towards the target frame origin $p_f^W(t)$:

$$\Delta\theta_{target} = \arctan\left(\frac{y_f^W - y_i^W}{x_f^W - x_i^W}\right) - \theta_i. \tag{1}$$

Here and henceforth, we will use, to simplify the notation, $p(t) = p_i^W(t)$.
The Freezone algorithm steps are:

1. At each time t create a polar diagram centered with the origin $p(t)$ of $\{R\}$, using the information of the distance to obstacles obtained from the sensors.
2. Find in this diagram all the possible candidate angles that will let the robot move safely. The candidates are the "valleys" of the diagram that allow the safe passage of the robot without bumping into obstacles. To find "valleys"

in the diagram a mask was created. This mask represents an angular interval with a minimum distance to the obstacles that allow the robot to pass between them.

3. Choose, among the candidates, the one that minimizes the initial relative heading $\Delta\theta_{target}$ w.r.t. the line connecting p_i^W to p_f^W subtarget, should the robot head towards that valley.
4. The safety condition of the robot is analyzed, and described as Low Safety or High Safety. Low Safety occurs if there is an obstacle inside a circular area, centered on the robot, with radius d_{max}. High Safety occurs otherwise.
5. During the robot motion, the desired rotation $\Delta\theta(t)$ suggests the reference for the rotational velocity to be:

$$\omega(t) = \begin{cases} -\omega_{max} & \Delta\theta(t) < -\frac{\pi}{2} \\ \omega_{max}\frac{\Delta\theta(t)}{\pi/2} & -\frac{\pi}{2} \leq \Delta\theta(t) \leq \frac{\pi}{2} \\ \omega_{max} & \Delta\theta(t) > \frac{\pi}{2} \end{cases} \tag{2}$$

where ω_{max} is the maximum rotational speed of the robot;

6. The safety condition of the robot is used to choose between two control laws for the translational velocity of the robot:

$$v(t) = \begin{cases} v_{max}\left(\frac{d_{obs}(t)}{d_{max}}\right)\left(1 - \left|\frac{\Delta\theta(t)}{\pi/2}\right|\right) & \text{If Low Safety} \\ \\ v_{max}\left(1 - \left|\frac{\Delta\theta(t)}{\pi/2}\right|\right) & \text{If High Safety} \end{cases} \tag{3}$$

where:

v_{max} is the maximum speed of the robot;
d_{obs} is the distance to the closest obstacle;
$\Delta\theta$ is the desired rotation angle of the robot.

The first equation has two terms that decrease the velocity in the presence of an obstacle (linear velocity decreases with the distance to the obstacle) and when turning. A saturation was imposed to the rotation of the robot, $\Delta\theta \in [-\pi/2, \pi/2]$. At the saturation limits, the robot must turn with maximum angular velocity, and no translational velocity, thus leading to quick turns towards the desired heading.

3 Freezone Applied to Soccer Robots

This algorithm was implemented in all the Nomadic SuperScout II robots of the ISocRob team. The sensors used for obstacle avoidance were the robot sonars, while odometry was used for guidance, reset after the occurrence of some critical situations (e.g., after bumping) by the vision-based self-localization method described in [5]. The implementation of the Freezone algorithm in the robots was as follows.

Model Identification All the model parameters were determined, from the safety region limits to the maximum rotational and translational speeds, as well as the maximum robot acceleration, and the conditions that ensure a safe path. The resolution of the sectors is 22.5 degrees centered with each sonar, being the geometric configuration of the sonars used as sectors $S = \{s_1, s_2, ..., s_{17}\}$. For example s_9 is the sector of the sonar located at the front of the robot. We also denote S_{ri} as the reading of the sector (sonar) i.

One of the restrictions considered in the sector configuration used was the robot geometry. The robot is a cylinder with 20.7 cm of radius and a kicker device 14 cm long (l_k) and 24 cm wide (w_k). The robot must be allowed to rotate near a wall. Another restriction was that the robot must keep its distance to a wall without steering. This restriction is very important to solve one of the problems of the Potential Fields method. Without this restriction, the robot, while following a wall, starts to oscillate. Imposing the restriction it is possible to maintain the velocity of the robot when near to a wall or obstacles, even when the angle between the target and the robot is more than 90 degrees.

For the first restriction the maximum distance a from the center of the robot to the outmost point of the robot was measured, and this distance was considered to be the minimum distance (d_{min}) from a wall, so that the robot is allowed to rotate near a wall. For the robots used the distance was $d_{min} = 35cm$.

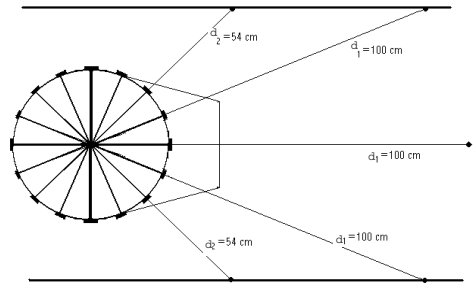


Fig. 2. Expected sonar measurements when traveling near a wall.

The expected sonar measurements when traveling near a wall were computed. These are shown in Fig. 2.

To compute the safety regions, some parameters must be determined first: the maximum acceleration, velocity and sample time of one sonar cycle. From the manufacturer's data, the robot maximum acceleration is a_{max} =0.8 m/s^2, the maximum velocity used is v_{max} =0.6 m/s and the sonar sample time is $T_{sample} = 0.032$ s.

At v_{max}, the robot needs to break until it stops, within a distance $d_{vmaxstop}$ given by:

$$d_{vmaxstop} = \frac{v_{max}^2}{2a_{max}}. \tag{4}$$

The distance $d_{samplesonars}$ traversed by robot with velocity v_{max} during one sonar cycle is given by

$$d_{samplesonars} = v_{max} \times T_{sample}. \tag{5}$$

The Minimum Safety Distance $d_{Safemin}$ when the robot moves at v_{max} is thus given by

$$d_{Safemin} = d_{min} + d_{vmaxstop} + d_{samplesonars}. \tag{6}$$

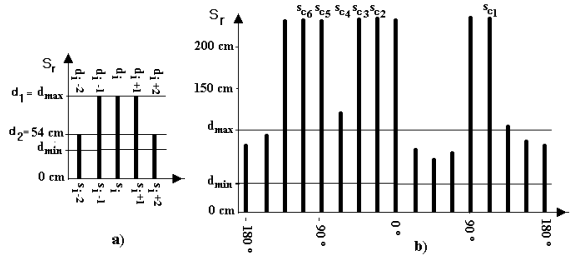


Fig. 3. a) Mask with the geometric configuration of the sectors; b) Typical Polar Histogram, showing the set S_c of sector candidates.

In this implementation, d_{min}=35 cm, $d_{vmaxstop}$=22.5 cm , $d_{samplesonars}$=2 cm hence $d_{Safemin}$=59.5 cm.

The $d_{Safemin}$ was used to evaluate the geometric sector configuration of the sectors used and to find the Maximum Safe Distance d_{max} (which must be greater than or equal to $d_{Safemin}$) inside which the obstacles are considered. In this implementation d_{max}=100 cm. In Fig. 2, d_1 and d_2 are the calculated distances from the robot center to the walls, read by the sonars s_{i-2}, s_{i-1}, s_i, s_{i+1} and s_{i+2}.

After identifying the model, the algorithm proceeds as explained in the following subsections.

Polar Diagram A polar diagram is created with the information given from the robot sonar sensors, as described in step 1 of section 2. In this implementation, the polar diagram is composed of the information given by the sonars only, with an accuracy of 22,5°, from -180° up to 180°, where 0° is the angle corresponding to the velocity axis of the robot. Figure 3.b) shows a typical robot polar histogram.

Find the Possible Candidates For model identification purposes, the configuration used for the sectors was computed (see Fig. 2), and a mask with those sectors was created, as shown in Fig. 3.a).

This mask was used to determine in which directions of S the robot can travel without endangering itself, as described in step 2 of Section 2. The algorithm uses the mask in the polar diagram, to find all the safe directions for the robot passage (step 2 of Section 2). The safety directions are collected into a vector $S_c = (s_{c1}, .., s_{cN}) \subseteq S$, where s_{ci} is a sector given by the mask. The pseudo code to use this mask is shown in Table 1.

Notice that Minguez et al.[6] and Borenstein[3], collect the valley extreme sectors (the right and left sectors). The near extreme is found and a new far extreme computed as the sum of the near border plus S_{max} (the angular interval that allows the passage of the robot), finding the desired steering direction $\Delta\theta$ from $\Delta\theta = (K_{nearborder} + K_{farborder})/2$, where $K_{farborder}$ and $K_{nearborder}$ are the angles between θ and the edge of the farthest and nearest borders of the valley. In the Freezone method, the direction is given by one of the orientation candidates.

for (i=2; i<15; i++)
{
 if (($s_{r(i-2)} \geq d_{i-2}$) and ($s_{r(i-1)} \geq d_{i-1}$) and ($s_{r(i)} \geq d_i$) and
  ($s_{r(i+1)} \geq d_{i+1}$) and ($s_{r(i+2)} \geq d_{i+2}$)) then
  {
   The direction s_i is a Safe Direction in S_c ;
  }
}

Table 1. Mask Pseudo code.

Choose the Orientation The orientation of safe travel selected is the sector from S_c that minimizes the direction to the target, with two exceptions that are presented at the end of this section, under the Special Case Conditions subsection.

The required rotation is designated as $\Delta\theta$ and can be expressed as:

$$\Delta\theta = \arg \min_{s_{ci} \epsilon S_c} \{|s_{ci} - \Delta\theta_{target}|, i = 1, 2, ..., N\}. \quad (7)$$

where $\Delta\theta_{target}$ is computed by (1). In this case, f refers to the goal target. This rotation provides a reference for the rotational velocity (see (2)).

Safety Conditions The safety condition (step 4 of Section 2) of the robot is analyzed. The robot is considered in High Safety if no obstacle is inside d_{max}=100

cm around the robot, and Low Safety if one or more obstacles are inside the safety region.

Translation Velocity As referred in step 6 of Section 2, the safety condition of the robot is used to choose between two alternative control laws of Section 2 for the translational velocity of the robot.

Special Case Conditions There are some cases where the algorithm leads to oscillations. Two of the most common are the following:

When the robot has a wall on its way to the goal posture and the wall is perpendicular to the trajectory of the robot to the goal. In this case the robot will rotate to one of the sides, and the angle between the goal and the direction of traveling will become > 90 degrees. At some point the robot will stop and rotate back 180 degrees, returning to the same point where it first arrived and the process will be repeated.

When a narrow obstacle is in between the target and the robot. In this case the robot will try to turn towards one of the obstacle sides. However, while turning, the angle to the target will increase and the the robot will eventually choose to go to the other side of the obstacle. This process will create an oscillatory motion that will make the robot collide with the obstacle.

To solve these two problems a new condition was introduced. If there is an obstacle in the direction of the target, in a distance less than d_{minobs}, the safe travel orientation selected is the one that minimizes the robot steering, aligning the robot with the direction $\Delta\theta(t-1)$, corresponding to the last reference for the steering angle.

$$\Delta\theta(t) = \arg \min_{s_{ci} \in S_c} \{|s_{ci} - \Delta\theta(t-1)|, i = 1, 2, ..., N\}. \tag{8}$$

4 Experimental Tests

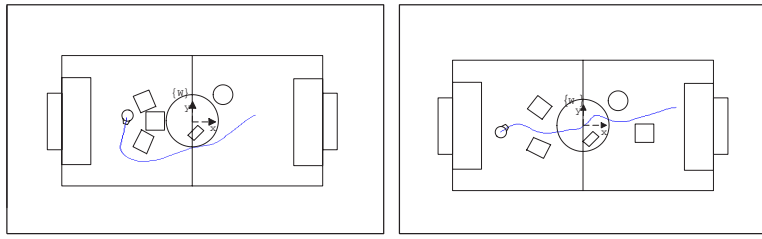
The tests presented with the Freezone method illustrate the resolution of the Potential Fields algorithm problems. The path shown was obtained from the robot odometry and the initial position of each run starts by determining the robot posture using the self-localization method. Notice that robot posture was determined with obstacles surrounding him. The robot's maximum speed was set to 0.6 m/s, $d_{max} = 100$ cm and $d_{min} = 35$ cm .

4.1 Test of Trap Situations Due to Local Minima

In these tests, some obstacles were placed inside the field (the four rectangles in the figures are boxes and the circle is the representation of a robot), and form a V-shaped obstacle between the robot and the goal target. This test allows to find the distance d_{minobs} that must be used to determine if there is an obstacle

between the robot and the direct path to the target posture. This distance can be used to solve the local minima of a U or V-shaped obstacle.

When using a minimum distance of only d_{minobs} =10 cm from the robot to an obstacle, the robot oscillates inside the V-shaped obstacle. In the test depicted in Fig. 4.a), the robot made a smooth trajectory from inside the V obstacle to the target posture, using d_{minobs} =40 cm.



a) b)

Fig. 4. a) Test of trap situations using a distance of d_{minobs} =40 cm. b)Smooth passage between closely spaced obstacles. $\xi_i = (-3, 0.5, 45°)$ and $\xi_f = (3, 1, 0°)$.

4.2 The Passage between Closely Spaced Obstacles

In these tests some of the obstacles were placed near each other and the robot had to choose either to pass through or to go around the obstacles. Notice that the two boxes in the left of the Figure 4.b), are 70 cm apart (closer point), while the robot and the box of the right field are separated by only 60 cm (closer point). In the following tests, the initial robot posture was changed in order to analyze different cases. It is possible to notice that the robot made a smooth path between the obstacles.

4.3 Oscillations in the Presence of Obstacles

The robot always tries to keep a certain distance from the obstacle, as seen in previous tests, and, because it is always looking for valleys, the path trajectory will be smooth, except if an obstacle is placed in front of the robot, in which case it turns towards one of the directions. However, in the presence of an obstacle and increasing the steering angle, the translational velocity is reduced to almost zero, making the robot stop to decide where to go.

5 Conclusions

A novel method for guidance with obstacle avoidance was implemented in the team robots : the Freezone. The Freezone algorithm was chosen to be used in

the ISocRob robots state-machine, due to the superior results obtained when compared to the Potential Fields methods, namely higher velocity and smoother trajectories while avoiding the obstacles in between the initial posture and the target posture. This method was used to place the robots in different postures, such as facing the goal with the ball in between or going to its home position, avoiding obstacles in between. Future work to improve the guidance with obstacle avoidance of the robots include: introduction of a certainty grid or histogram grid to include more information or uncertainty of the sonar data, adaptation of the Freezone method for situations when the robot has the ball to allow control of the ball by controlling the path geometry and velocity, sensor fusion, using both the sonars and the catadioptric system to identify obstacles.

References

1. Baltes, J., and Hildreth, N., "Adaptive Path Planner for Highly Dynamic Environments", RoboCup-2000: Robot Soccer World Cup IV, Springer Verlag, pp. 76-85, Berlin, 2001
2. Borenstein, J., and Koren, Y., "Potential Fields Methods and Their Inherent Limitactions for Mobile Navigation", Proceedings of the IEEE International Conference on Robotics and Automation, Sacramento, California, pp. 1398-1404, April 7-12, 1991
3. Borenstein, J., "Real-time Obstacle Avoidance for Fast Mobile Robots", IEEE Transactions on Systems, Man and Cybernetics, Vol. 19, No. 5, pp. 1179-1187, Sept./Oct. 1989
4. Krogh, B. H., "A Generalized Potencial Field Approach to Obstacle Avoindance Control", Proceedings of International Robotics Research Conference, Bethlehem, Pennsylvania, August, 1984
5. Marques, C. and Lima, P., "A Localization Method for a soccer Robot using a Vision-Based Omni-directional sensor", RoboCup-2000: Robot Soccer World Cup IV, Springer Verlag, pp. 96-107, Berlin, 2001
6. Minguez, J. and Montano, L., "Nearness Diagram Navigation(ND): A New Real Time Collision Avoidance Approach.",Proceedings IEEE/IROS2000, Takamatsu, Japan, Oct.30-Nov.5, 2000
7. Nagasaka, Y., Murakami, K., Naruse, T., Takahashi, T., Mori, Y., "Potential Field Approach to Short Term Action Planning in RoboCup F180 League", RoboCup-2000: Robot Soccer World Cup IV, Springer Verlag, pp. 345-350, Berlin, 2001
8. Khatib, O., "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots." 1985 IEEE International Conference on Robotics and Automation, St. Louis, Missouri, pp. 500-505, March 25-28, 1985
9. Canudas de Wit, C., Siciliano, B., Bastin, G. (Eds), "Theory of Robot Control", CCE Series, Kluwer, 1996
10. Minoru Asada, Tucker Balch, Raffaelo D'Andrea, Masahiro Fujita, Bernhard Hengst, Gerhald Kraetzschmar, Pedro Lima, Nuno Lau, Henrik Lund, Daniel Polani, Paul Scerri, Satoshi Tadakoro, Thilo Weigel, Gordon Wyeth, "RoboCup-2000: The Fourth Robotic Soccer World Championships", AI-Magazine, volume 22, NO.1, Spring 2001