World modeling on an MSL robotic soccer team

Article *in* Mechatronics · March 2011

DOI: 10.1016/j.mechatronics.2010.05.011

5 authors, including:

João Silva
University of Aveiro
14 PUBLICATIONS **68** CITATIONS
SEE PROFILE

Nuno Lau
University of Aveiro
211 PUBLICATIONS **1,348** CITATIONS
SEE PROFILE

António J. R. Neves
University of Aveiro
162 PUBLICATIONS **730** CITATIONS
SEE PROFILE

João Manuel Rodrigues
University of Aveiro
60 PUBLICATIONS **346** CITATIONS
SEE PROFILE

Mechatronics xxx (2010) xxx–xxx

Contents lists available at ScienceDirect

Mechatronics

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

World modeling on an MSL robotic soccer team

João Silva *, Nuno Lau, António J.R. Neves, João Rodrigues, José Luís Azevedo

ATRI, IEETA/DETI, University of Aveiro, 3810-193 Aveiro, Portugal

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Sensor fusion
World model
Kalman filter
Linear regression
Obstacle detection
Visual matching

ABSTRACT

When a team of robots is built with the objective of playing soccer, the coordination and control algorithms must reason, decide and actuate based on the current conditions of the robot and its surroundings. This is where sensor and information fusion techniques appear, providing the means to build an accurate model of the world around the robot, based on its own limited sensor information and the also limited information obtained through communication with the team mates. One of the most important elements of the world model is the robot self-localization, as to be able to decide what to do in an effective way, it must know its position in the field of play. In this paper, the team localization algorithm is presented focusing on the integration of visual and compass information. An important element in a soccer game, perhaps the most important, is the ball. To improve the estimations of the ball position and velocity, two different techniques have been developed. A study of the visual sensor noise is presented and, according to this analysis, the resulting noise variation is used to define the parameters of a Kalman filter for ball position estimation. Moreover, linear regression is used for velocity estimation purposes, both for the ball and the robot. This implementation of linear regression has an adaptive buffer size so that, on hard deviations from the path (detected using the Kalman filter), the regression converges faster. A team cooperation method based on sharing the ball position is presented. Other important data during the soccer game is obstacle data. This is an important challenge for cooperation purposes, allowing the improvement of team strategy with ball covering, dribble corridor estimation, pass lines, among other strategic possibilities. Thus, detecting the obstacles is ceasing to be enough and identifying which obstacles are team mates and opponents is becoming a need. An approach for this identification is presented, considering the visual information, the known characteristics of the team robots and shared localization among team members. The described work was implemented on the CAMBADA team and allowed it to achieve particularly good performances in the last two years, with a 1st and a 3rd place in the world championship RoboCup 2008 and RoboCup 2009 editions, respectively, as well as distinctively achieve 1st place in 2008 and 2009 editions of the Portuguese Robotics Open.

1. Introduction

Nowadays, there are several research domains in the area of multi robot systems. One of the most popular is robotic soccer. RoboCup[1] is an international joint project to promote artificial intelligence, robotics and related fields. Most of the RoboCup leagues have soccer as platform for developing technology, either at software or hardware levels, with single or multiple agents, cooperative or competitive [1].

Among RoboCup leagues, the Middle Size League (MSL) is one of the most challenging. In this league, each team is composed of up to five robots with maximum size of 50 × 50 cm base, 80 cm height and a maximum weight of 40 kg, playing in a field of 18 × 12 m. The rules of the game are similar to the official FIFA rules, with required changes to adapt for the playing robots [2].

Each robot is autonomous and has its own sensorial means. They can communicate with each other, and with an external computer acting as a coach, through a wireless network. This coach computer cannot have any sensor, it only knows what is reported by the playing robots. The agents should be able to evaluate the state of the world and make decisions suitable to fulfill the cooperative team objective.

CAMBADA, *Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture*, is the Middle Size League Robotic Soccer team from the University of Aveiro. The project started in 2003, coordinated by the IEETA[2] ATRI[3] group and involves people working

* Corresponding author.
E-mail addresses: joao.m.silva@ua.pt (J. Silva), nunolau@ua.pt (N. Lau), an@ua.pt (A.J.R. Neves), jmr@ua.pt (J. Rodrigues), jla@ua.pt (J.L. Azevedo).

[1] http://www.robocup.org/.

[2] Instituto de Engenharia Electrónica e Telemática de Aveiro – Aveiro's Institute of Electronic and Telematic Engineering.

[3] Actividade Transversal em Robótica Inteligente – Transverse Activity on Intelligent Robotics.

0957-4158/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mechatronics.2010.05.011

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

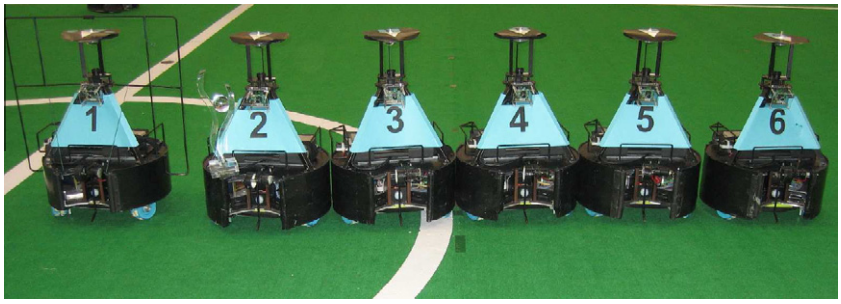

Fig. 1. Picture of the team robots used to obtain the results presented on this paper.

on several areas for building the mechanical structure of the robot, its hardware architecture and controllers and the software development in areas such as image analysis and processing, sensor and information fusion, reasoning and control (see Fig. 1).

This paper provides a description of some sensor and information fusion techniques and algorithms used in the CAMBADA team. The data obtained by these techniques are necessary for building a world model of the robot environment. This paper includes the description of some of the elements of that model necessary for a team of robots to play soccer. In Section 2, a brief overview of some related topics and work in sensor and information fusion for world modeling are presented. Section 3 presents the team self-localization description, introducing it as the first necessary step for all the other information fusion. In Section 4, the ball integration process is presented in all its components, starting with the ball position, its velocity and finally its sharing among team mates. Section 5 presents an overview of obstacle treatment, with some visual detection details, the matching of positions for visual identification and the sharing of information among team mates. Finally, Section 6 concludes the paper.

2. Related work

World modeling and sensor and information fusion are tightly related, as the latest provide the means to build the desired model. Sensor and information fusion is the process of combining sensory data, or data derived from sensory data, providing a resulting information that is better than would be possible when the sources were used individually [3]. One of the main areas where sensor fusion techniques are used is position tracking, both for self and object localization/tracking.

The integration of information over time in order to filter sensor noise is essential to get better estimates. This type of integration may be performed using Kalman filter based approaches, Monte Carlo methods or Markov approaches. Generally, Monte Carlo [4] approaches have better performance in cases where great discontinuities of the output values are expected, as the assumption of Gaussian probability density functions of the Kalman filter [5] is usually less accurate. However, Kalman filtering is a very effective method if the assumptions of Gaussian noise can be met and the system can be linearized. Other common approaches are the use of the Extended and Unscented Kalman filters [6], which are prepared to deal with non-linear systems at the cost of more computational weight.

A general overview of different methods of multi-sensor and information fusion is presented in [7], also with a brief description of application areas, such as robotics, military, biomedical and transportation. Applications in the robotics field include self-localization using either Kalman filter [8], Monte Carlo [9] or Markov [10] methods, or integration of information coming from several robots, to increase the accuracy of each of the robots position estimation [11]. A general recent overview of methods and architectures for multi-sensor data fusion can be found in [12].

Another recurrent problem nowadays is the fusion of visual and inertial sensors [13], where recent results have demonstrated that the visual tracking of objects may work at higher velocities and be more robust if combined with information coming from inertial sensors [14] and also that ego-motion estimation can be more precise and navigation more robust using these approaches [15].

Simultaneous Localization And Mapping (SLAM) is another common application of sensor fusion techniques, as in many cases, autonomous robots have to map the environment rather than simply localize themselves [16,17].

Particularly in RoboCup domain, several teams use this kind of approaches, not only for localization purposes, but also for position estimation and tracking of objects, namely the ball and other robots. Several teams have used Kalman filters for the ball position estimation [18–21]. In [20,21], several information fusion methods are compared for the integration of the ball position using several observers. In [21], the authors conclude that the Kalman reset filter shows the best performance.

Although using well known techniques, in this paper we propose practical solutions for an efficient self-localization, ball information treatment and obstacle treatment for an MSL robotic soccer team. As far as we know, no previous work has been published focusing on these several important aspects of developing the world model of an MSL soccer team.

3. Localization

Self-localization of the agent is an important issue for a soccer team, as strategic moves and positioning must be defined by positions on the field. In the MSL, the environment is partially known, as every agent knows exactly the layout of the game field but does not know the position of any other elements, either itself, other robots or the ball. Given the known map, the agent has then to locate itself.

The CAMBADA team localization algorithm is based on the detected field lines, with fusion of information from the odometry sensors and an electronic compass. It is based on the approach described in [22], with some adaptations. It can be seen as an error minimization task, with a derived measure of reliability of the calculated position so that a stochastic sensor fusion process can be applied to increase the estimation accuracy [22].

From the center of the image (the center of the robot), radial sensors are created around the robot, each one represented by a line with a given angle. These are called *scanlines*. The image processing, in each cycle, returns a list of positions relative to the robot where the *scanlines* intercept the field line markings [23].

The idea is to analyze the detected line points, estimating a position, and through an error function describe the fitness of the estimation. This is done by reducing the error of the matching

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

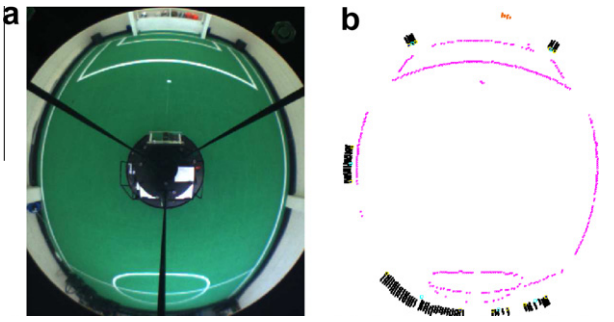

Fig. 2. Captures of an image acquired by the robot camera and processed by the vision algorithms. Left (a): The image acquired by the camera. Right (b): The same image after processing with magenta dots over the detected field lines.

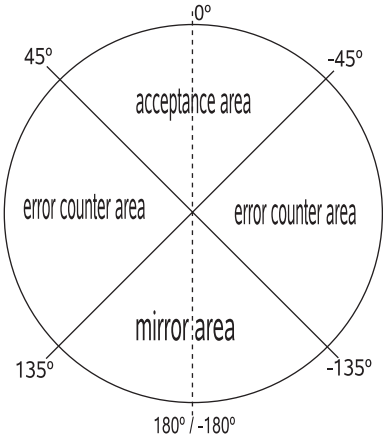

Fig. 3. Illustration of the compass error angle intervals.

between the detected lines and the known field lines (Fig. 2). The error function must be defined considering the substantial amount of noise that affects the detected line points which would distort the representation estimation [22].

In normal operation mode, the localization is done over a limited set of base positions from which tracking is maintained. Since it is an algorithm based on optimization and since there are many local minima, the tracking only works satisfactorily if the estimations are near the solution. In situations where the robot does not possess a valid estimation, a global localization algorithm estimates the robot position on the field using a much wider set of initial estimations over which the already referred error minimization process for optimization is applied. However, this global localization algorithm is computationally heavy and time consuming. For that reason, after having an initial position, the simpler tracking localization handles the cyclic relocation.

Although the odometry measurement quality quickly degrades with time, within the reduced cycle times achieved in the application, consecutive readings produce acceptable results and thus, having the visual estimation, it is fused with the odometry values to refine the estimation. This fusion is based on a Kalman filter for the robot position estimated by odometry and the robot position estimated by visual information. This approach allows the agent to estimate its position even if no visual information is available. However, it is not reliable to use only odometry values to estimate the position for more than a few cycles, as slidings and frictions on the wheels produce large errors on the estimations in short time.

Due to the nature of the approach, this algorithm works acceptably with a relatively low number of points, like a few tens of points, as long as they are representative of the surroundings. Consider the case of matching a 90 degrees corner. If the algorithm had access to 200 points all over the same line, it would not be capable of matching the corner. On the other hand, with only 20 or 30 points scattered over both lines, the algorithm would be capable

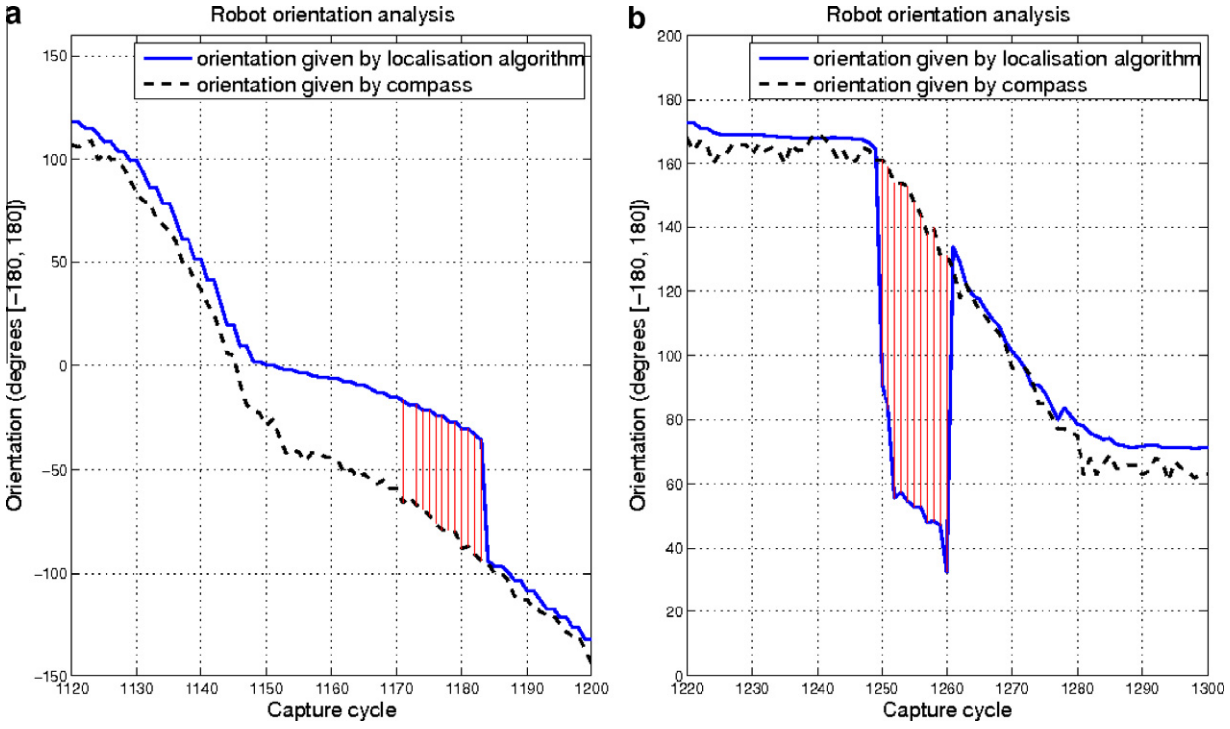

Fig. 4. Illustration of two situations where relocation was forced. Dashed line represents the angle given by the compass, solid line represents the angle estimated by the localization algorithm, red lines represent the cycles on which the error between the two angles is greater than the threshold. Left (a): The camera was covered while the robot moved. The estimated orientation error degrades progressively and after getting higher than a threshold, the cycle count starts and forces relocation. Right (b): The robot tilted. The estimated orientation error is immediately affected by more than a threshold and the cycle count starts and forces relocation.

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

of detecting the match. Even in situations where the points are over the same line, the merging with odometry and position tracking provide a good robustness to the algorithm [22], as long as the situation is temporary, which is usually the case.

The visually estimated orientation can be ambiguous, i.e. each point on the soccer field has a symmetric position, relatively to the field center, where the robot detects exactly the same field lines. To disambiguate the symmetry problem and to detect wrong estimations, an electronic compass is used. The orientation estimated by the robot is compared to the orientation given by the compass and if the error between them is larger than a predefined threshold, actions are taken. If the error is really large (i.e. around ±180 degrees), it means that the robot estimated orientation is symmetric to the real one, so it should assume the mirror position. On the other hand, if the error is larger than the acceptance threshold (i.e. a 90 degrees acceptable area), a counter is incremented (Fig. 3).

This counter will be incremented every cycle in which the error is greater than the threshold. If a given number of consecutive cycles with high errors is reached (i.e. the counter reaches a given number, currently 10), the robot considers itself "lost", meaning that it will not continue to track its position but will instead consider the initial situation, with no *a priori* knowledge and thus executes the global localization algorithm. Fig. 4 shows situations where the threshold was reached and relocation was forced after some cycles.

4. Ball integration

The information of the ball state (position and velocity) is, perhaps, the most important, as it is the main object of the game and it is the base over which most decisions are taken. Thus, its integration has to be as reliable as possible. To accomplish this, a Kalman filter implementation was created to filter the estimated ball position given by the visual information, and a linear regression was applied over filtered positions to estimate its velocity.

4.1. Ball position

It is assumed that the ball velocity is constant between cycles. Although that is not true, due to the short time variations between cycles, around 40 ms, and given the noisy environment and measurement errors, it is a quite acceptable model for the ball movement. Thus, no friction is considered to affect the ball, and the model does not include any kind of control over the ball. Therefore, given the Kalman filter formulation (described in [24]), the assumed state transition model is given by

$$X_k = \begin{bmatrix} 1 & \Delta T \\ 0 & 1 \end{bmatrix} X_{k-1}$$

where $X_k = \begin{bmatrix} Pos \\ Vel \end{bmatrix}$ is the state vector containing the position and velocity of the ball. Both are composed by the respective (x, y) coordinates. This velocity is only internally estimated by the filter, as the robot sensors can only take measurements on the ball position. After defining the state transition model based on the ball movement assumptions described above and the observation model, the description of the measurements and process noises are important issues to attend. The measurements noise can be statistically estimated by taking measurements of a static ball position at known distances.

In practice, measurements of the static ball were taken while the robot was rotating around its vertical axis and this was done with the ball placed at several distances, measured with metric tape. Although real game conditions are probably more adverse, we lack the means to externally know the position of the elements on the field. For that reason, to know the real distance between the robot and the ball, we opted to use the described setup. Some of the results are illustrated in Fig. 5.

The standard deviation of those measurements can be used to calculate the variance and thus define the measurements noise parameter.

A relation between the distance of the ball to the robot and the measurements standard deviation can be modeled by a 2nd degree polynomial best fitting the data set in a least-squares sense. Depending on the available data, a polynomial of another degree could be used, but we should always keep in mind the computational weight of increasing complexity.

As for the process noise, this is not trivial to estimate, since there is no way to take independent measurements of the process to estimate its standard deviation. The process noise is represented by a matrix containing the covariances correspondent to the state variable vector.

Based on the Kalman filter functioning, one can verify that forcing a near null process noise causes the filter to practically ignore the read measures, leading the filter to emphasize the model prediction. This makes it too smooth and therefore inappropriate. On the other hand, if it is too high, the read measures are taken too much into account and the filter returns the measures themselves.

To face this situation, one has to find a compromise between stability and reaction. Since we assume an uniform movement for the ball, there are no frictions or other external forces considered. This means that accelerations are not considered in our model and thus, the position and velocity components are quite independent of each other. Since acceleration is the main element of relation between position and velocity, we considered that the errors associated to the process position and velocity estimations do not correlate.

Because we assume an uniform movement model that we know is not the true nature of the system, we know that the speed calculation of the model is not very accurate. A process noise covariance

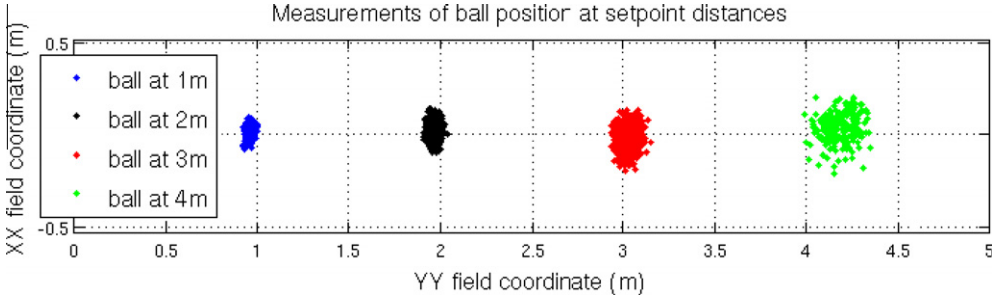

Fig. 5. Noisy position of a static ball taken from a rotating robot.

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

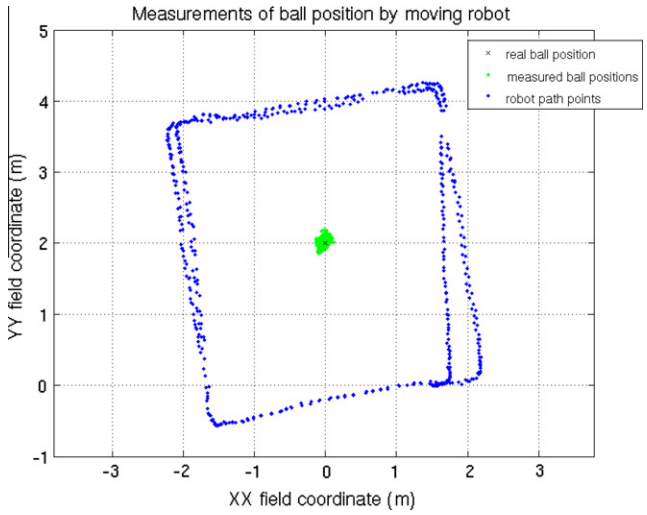

Fig. 6. Plot of a robot movement around a fixed ball position. The ball positions measured by the moving robot form a cloud of points (green) in the area of the real ball position (black X). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

matrix was empirically estimated, based on several tests, so that a good smoothness/reactivity relationship was kept. These empirically estimated values were made dependent on the measurement noise so that the Kalman filter predictions are also less accurate when the distance to the ball is too large. This was done so that the filter does not smooth the positions too much.

In practice, this approach proved to improve the estimation of the ball position. Since we do not possess the means to externally know the positions of the elements on the field, a capture was made with the ball fixed at a known position on the field $(0.0, 2.0)$ (measured with metric tape). The robot was moving around the ball with a speed of 1.3 ± 0.5 m/s and the ball position measured at each moment was recorded. The ball position measured by the robot was $(-0.01, 2.03) \pm (0.05, 0.06)$ m. Fig. 6 illustrates the capture results.

This experiment gives an idea of the noise associated with the ball position detection. Note that during the experiment the distance between the robot and the ball is around 2 m. Comparing the ball position cloud with the one obtained at 2 m in Fig. 5 one can verify that they are similar, which is consistent with the previous experiment setup to simulate robot movement by rotation on the spot.

With the presented setup experiments, the existence of noise in ball measurements became clear. With that existent noise in mind, several tests were made to validate the use of the Kalman filter to reduce it. Fig. 7 represents a capture of one of those tests, a ball movement, where the black dots are the ball positions measured by the robot visual sensors and thus are unfiltered. Red stars[4] represent the position estimations after applying the Kalman filter. The robot position is represented by the black star in its center and its respective radius. The ball was thrown against the robot and deviated accordingly. It is easily perceptible that the unfiltered positions are affected by much noise and the path of the ball after the collision is composed of positions that do not make much physical sense. Although we lack the means to externally provide a ground truth for the ball position during its movements, the filtered positions seem to give a much better approximation to the real path taken by the ball, as they provide a path that physically makes more sense.

[4] For interpretation of color in 'Figs. 1,2,4-7,9-11,13-15,17-20' the reader is referred to the web version of this article.

After producing the *a priori* estimation of the ball position, this estimation is compared with the read measure to detect if the variation between them is too great. If the difference between them is consistently greater than a given threshold (estimated empirically), the filter can indicate that the ball suffered a hard deviation (Fig. 8 illustrates this concept).

Although hard deviations are not a serious problem for the filter (as it quickly converges to the new positions), they are used for velocity convergence (as described in the next subsection).

4.2. Ball velocity

The calculation of the ball velocity is a feature becoming more and more important over the time. It allows that better decisions can be implemented based on the ball speed value and direction. Assuming the same ball movement model described before, constant ball velocity between cycles and no friction considered, one could theoretically calculate the ball velocity by simple instantaneous velocity of the ball with the first order derivative of each component $\frac{\Delta D}{\Delta T}$, being ΔD the displacement on consecutive measures and ΔT the time interval between consecutive measures. However, given the noisy environment, it is also predictable that this approach would be greatly affected by that noise and thus its results would not be satisfactory.

Fig. 9 shows a ball movement capture where the ball was moving from left to right, as indicated by the arrow in the top of the figure, and was then deviated into a downward movement near the "1st deviation" tag. While moving downward, the ball was deviated again near the "2nd deviation" tag and started to move from right to left. Finally, in the end of the capture, a new deviation occurred near tag "3rd deviation" where the ball started to move upward. The estimated ball positions are represented by the blue dots. Red lines represent the velocity vectors estimated based on consecutive positions displacement. It is clear that the velocity estimates hardly give an acceptable insight of the ball movement.

To keep a calculation of the object velocity consistent with its displacement, an implementation of a linear regression algorithm was chosen. This approach based on linear regression [25] is similar to the velocity estimation described in [18]. By keeping a buffer of the last m measures of the object position and sampling instant (in this case buffers of nine samples were used), one can calculate a regression line to fit the positions of the object. Since the object position is composed by two coordinates (x, y), we actually have two linear regression calculations, one for each dimension. This is made in a transparent way, so the description is presented generally, as if only one dimension was considered.

When applied over the positions estimation, the linear regression velocity estimations are much more accurate than the instant velocities calculated by $\frac{\Delta D}{\Delta T}$, and allow a better insight of the ball movement. The same ball movement capture described earlier is represented in Fig. 10, this time with the velocity vectors estimated by the linear regression applied over the position estimations provided by the Kalman filter.

In order to try to make the regression converge more quickly on deviations of the ball path, a reset feature was implemented. This allows deletion of the older values, keeping only the n most recent ones, and provides control of the buffer size. By keeping the most recent values after a hard deviation, we reduce outliers of the previous path, thus promoting faster convergence. This reset results from the interaction with the Kalman filter described earlier by querying it for the existence of a hard deviation on the ball path.

The obtained values were tested to confirm if the linear regression of the ball positions was more precise and would converge faster than the internal velocity estimated by the Kalman filter. Tests showed that the velocity estimated by the Kalman filter has a slower response than the linear regression estimation when

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

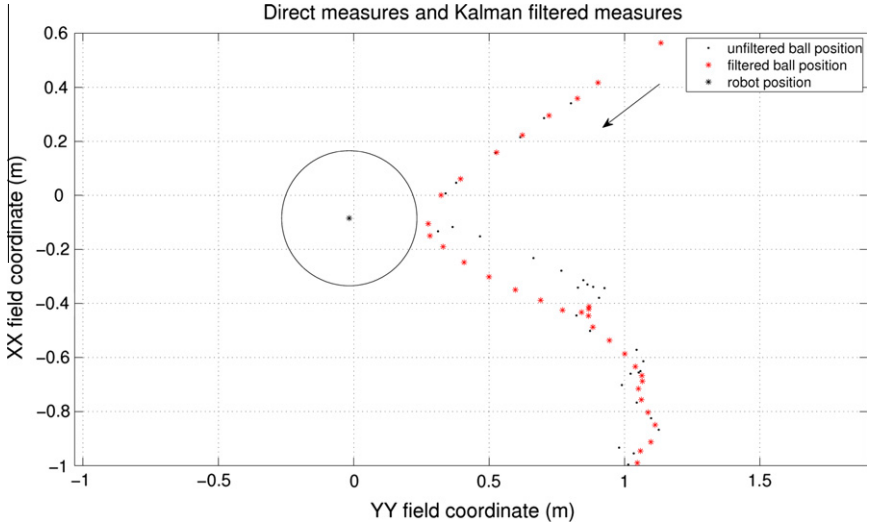

Fig. 7. Plot of a ball movement situation.

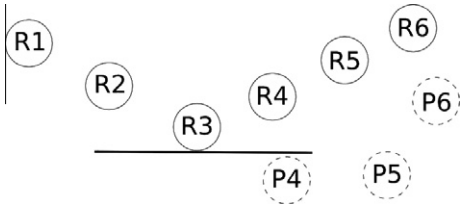

Fig. 8. Situation where a hard deviation would be detected by the filter. Positions R4,5,6, are the measured positions after the ball hits an obstacle, P4,5,6 are the predicted filtered estimations, which did not consider that something might alter the ball path.

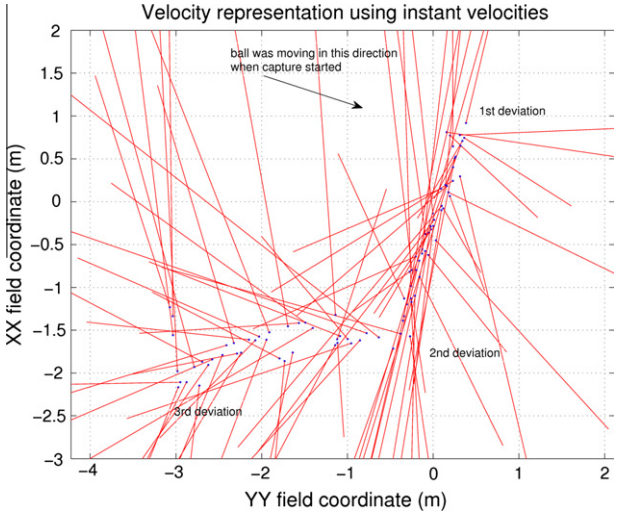

Fig. 9. Velocity representation using consecutive measures displacement.

deviations occur. Given this, the linear regression was used to estimate the velocity because quickness of convergence was preferred over the slightly smoother approximation of the Kalman filter in the steady state. That is because in the game environment the ball is very dynamic, it constantly changes its direction and thus a convergence in less than half the cycles is much preferred. Fig. 11 shows the results for a theoretical velocity scenario where the ball was moving at a constant speed of 2 m/s and suddenly dropped to

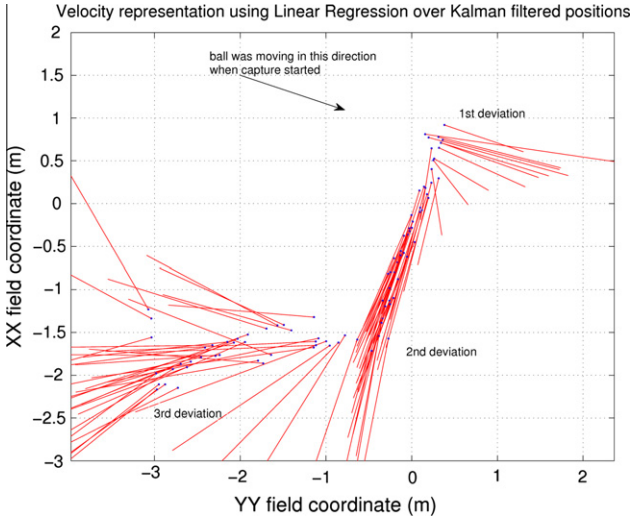

Fig. 10. Velocity representation using linear regression over Kalman filtered positions.

a constant 1 m/s speed. Both the speeds estimated by the Kalman filter and the ones estimated by the linear regression are presented.

4.3. Team ball position sharing

Due to the highly important role that the ball has in a soccer game, when a robot cannot detect it by its own visual sensors (omni or frontal camera), it may still know the position of the ball, through sharing of that knowledge by the other team mates.

The ball data structure includes a field with the number of cycles it was not visible by the robot, meaning that the ball position given by the vision sensors can be the "last seen" position. When the ball is not visible for more than a given number of cycles, the robot assumes that it cannot detect the ball on its own. When that is the case, it uses the information of the ball communicated by the other running team mates to know where the ball is. This can be done by getting the mean and standard deviation of the positions of the ball seen by team mates. Another approach is to simply use the ball position of the team mate that has more confidence in the detection.

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

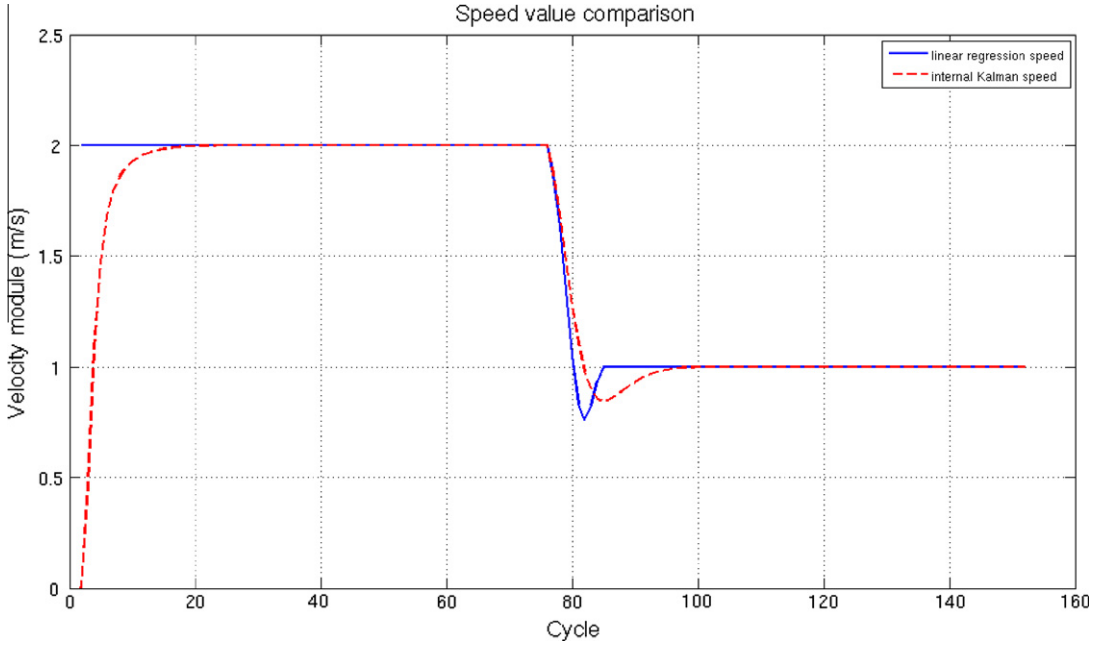

Fig. 11. Comparison between the velocity estimated by the linear regression (blue solid line, faster convergence) and internally by the Kalman filter (red dashed line, smoother, but of slow convergence). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Independently of the chosen approach, the robot assumes that ball position as correct. When detecting the ball on its own, there is also the need to validate that information.

Currently the seen ball is only considered if it is within a given margin inside the field of play as there would be no point in trying to play with a ball outside the field. For ball position sharing, an approach based on the highest confidence ball position is used. This is due to the fact that the shared positions are updated with 100 ms periods, with the possibility of a few more milliseconds of unknown and unpredictable delay in packet transmission. Thus, the lifetime of the information of each team mate is different, and the use of the information of the team mate with higher confidence reduces the probability of the degradation of that information during the respective lifetime. Fig. 12 illustrates the general ball integration activity diagram.

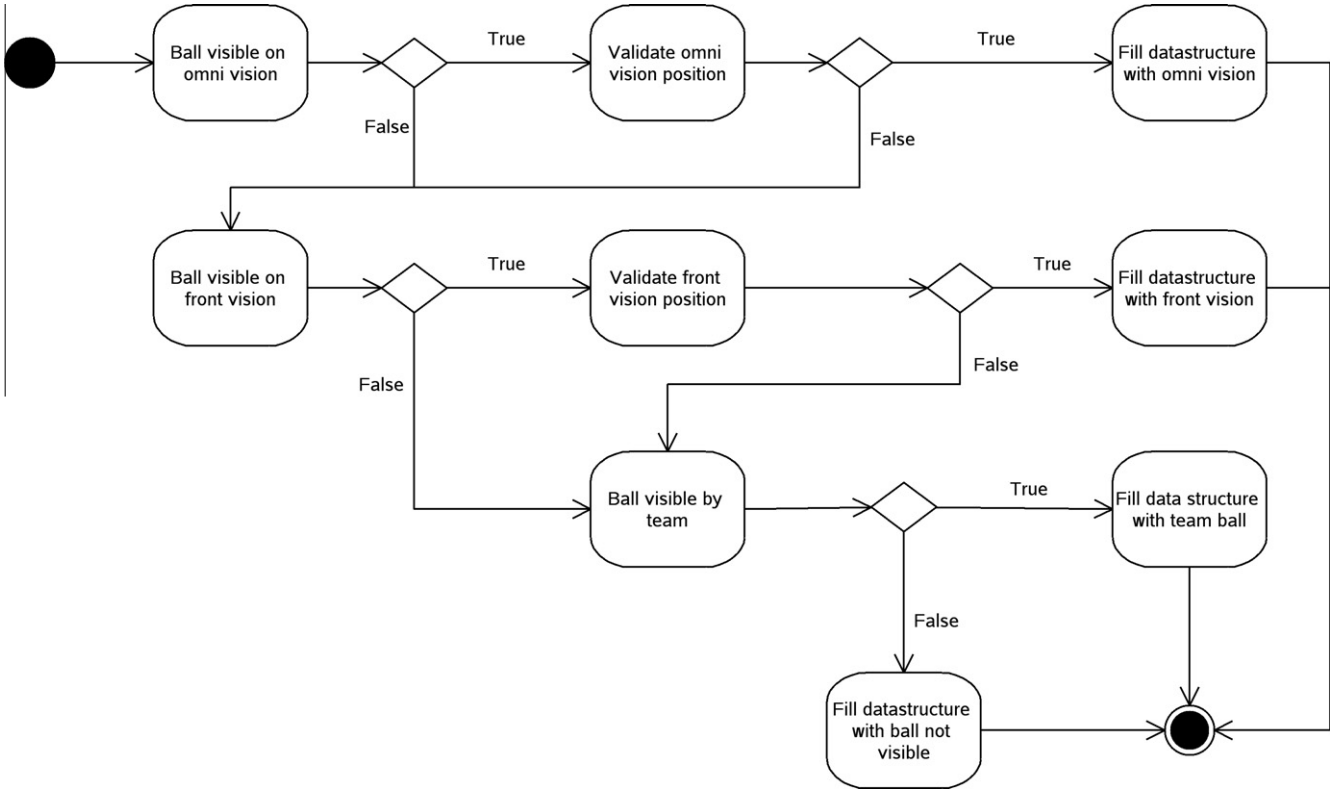

Fig. 12. Diagram of the ball integration algorithm.

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

5. Obstacle treatment

While playing soccer, the robots have the need to navigate around the field effectively, which means they have to reposition themselves or dribble the ball avoiding the obstacles on the field, that can be either team or opponent robots, or eventually the referee.

An increasing necessity felt by the team, to improve its performance, is a better obstacle detection and sharing of obstacle information among team mates. This is important to ensure a global idea of the field occupancy, since the team formation usually keeps the robots spread across the field. Pass lines and dribbling corridors can be estimated more easily with a good coverage of field obstacles, allowing improvements on team strategy and coordination.

5.1. Visual obstacle detection

The CAMBADA robots gather their information about the surroundings by means of a robotic vision system. Currently, only the omni directional camera gathers information about obstacles, as no frontal camera is being used at this time.

According to RoboCup rules, the robots are mainly black. Since during the game robots play autonomously, all obstacles in the field are the robots themselves (occasionally the referee, which is recommended to wear black/dark pants). The vision algorithm detects the obstacles by evaluating blobs of black color inside the field of play [26]. Through the mapping of image positions to real metric positions [27], obstacles are identified by their center (triangle on the processed image, Fig. 13b) and left and right limits (squares on the processed image, Fig. 13b). This is done by searching black regions on the *scanlines* of the vision algorithm [23], already referred in Section 3.

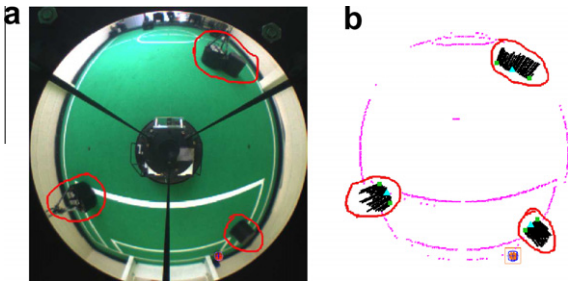

Fig. 13. Captures of an image acquired by the robot camera and processed by the vision algorithms. The areas of interest were surrounded. (a) The image acquired by the camera. (b) The same image after processing. Obstacles are identified by their center (triangle), left and right limits (squares). It is visible that the two aligned obstacles are detected as a single larger obstacle (top right of the frames).

The detection of black color on the scanlines is analyzed both in angular intervals and length intervals, to define the limits of each black blob (considering their base points which are represented by the first black pixel in each scanline). Since the vision system is a non-SVP hyperbolic catadioptric system [27], the size of objects on the image varies with the distance to the robot. Due to an inverse distance map calculation, by exploring a back-propagation ray-tracing approach and the geometric properties of the mirror surface, the relation of distances in the image and the real world is known. Fig. 14 is an illustration of how the distance in pixels, from the center of the image, is mapped to the distance in meters, on the ground plane.

Through the function represented in Fig. 14, it is possible to create a normalized relation of blobs width and length with the distance. Sometimes an obstacle is separated in several blobs, mainly due to the noise in the image and problems in color classification, which leads to failure in the detection of black regions in the scanlines. To avoid these situations, an offset is considered to decide when the angular space between blobs is considered enough to represent a real obstacle separation. The same principle is considered concerning the position of the black area in consecutive scanlines.

The separation offsets of a blob close to the robot are bigger than the ones at a high distance, to maintain coherent precision. The angular separation offset is considered for situations where robots are side-by-side, at the same distance, but there is no visual contact between each blob; the length separation offset is checked for situations where, on consecutive scanlines, there are blobs with visual contact but the robots are actually at different distances. Both situations are depicted in Fig. 15.

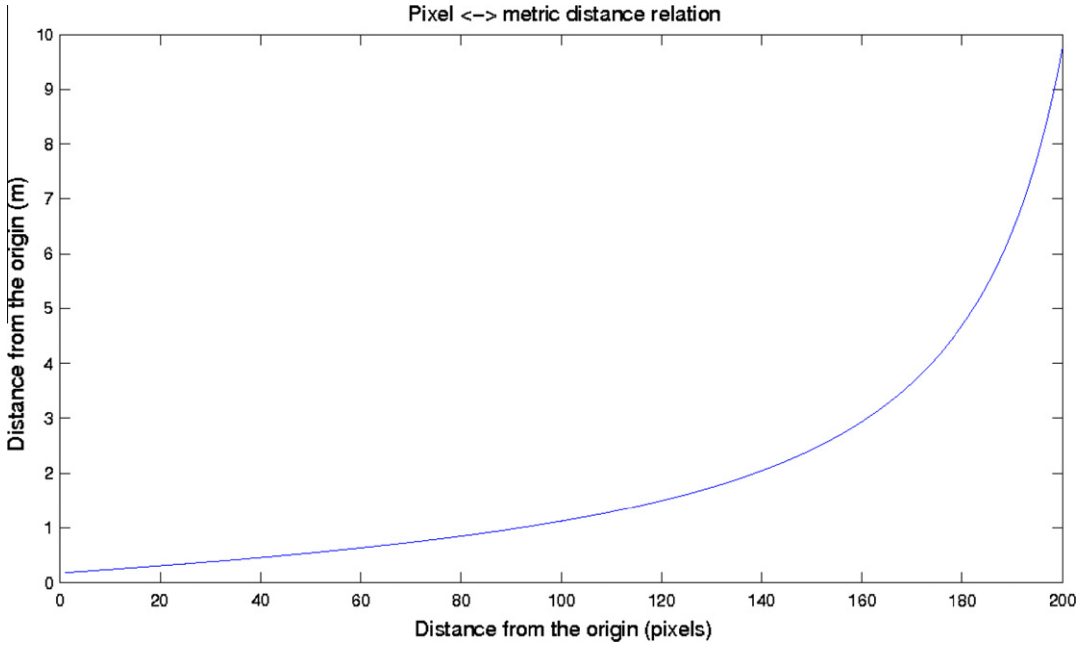

Fig. 14. Relation between pixels and metric distances. The center of the robot is considered the origin and the metric distances are considered on the ground plane.

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

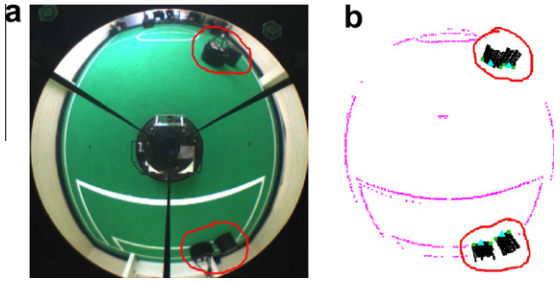

Fig. 15. Example of an image acquired by the robot camera and processed by the vision algorithm. The areas of interest are surrounded. (a) The image acquired by the camera. (b) The same image after processing. It is visible the two possibilities of separation made: angular separation, on the bottom pair of obstacles and length separation, on the top pair of obstacles.

For each detected blob, their number of pixels is calculated and an estimation of the obstacles left and right limits, as well as their centers, is made. This information is made available to the integration process for filtering and treatment.

5.2. Obstacle selection and identification

With the objective of refining the information of the obstacles, and have more meaningful and human readable information, the obstacles are selected and a matching is attempted, in order to try to identify them as team mates or opponents.

Due to the weak precision at long distances, a first selection of the obstacles is made by selecting only the obstacles closer than a given distance as available for identification (currently 5 m). Also, obstacles that are smaller than 10 cm wide or outside the field of play margin are ignored. This is done because the MSL robots are rather big, and in-game situations small obstacles are not present inside the field. Also, it would be pointless to pay attention to obstacles that are outside the field of play, since the surrounding environment is completely ignorable for the game development.

To be able to distinguish obstacles, identifying which of them are team mates and which are opponent robots, a fusion between the own visual information of the obstacles and the shared team mates positions is made. By creating a circle around the team mate positions with the robot radius (considered 22 cm), a matching of the estimated center of visible obstacle area is made (Fig. 16), and the obstacle is identified as the corresponding team mate in case of a positive matching (Figs. 17c and 18c). This matching consists on the existence of interception points between the team mate circle and the obstacle circle or if the obstacle center is inside the team mate circle (the obstacle circle can be smaller, and thus no interception points would exist).

Since the detected obstacles can be large blobs, the above described identification algorithm cannot be applied directly to the visually detected obstacles. If the detected obstacle fulfills the minimum size requisites already described, it is selected as candidate for being a robot obstacle. Its size is evaluated and classified as robot if it does not exceed the maximum size allowed for MSL robots [2] (Fig. 17a and b).

If the obstacle exceeds the maximum size of an MSL robot, a division of the obstacle is made, by analyzing its total size which is used to estimate how many robots are in that obstacle. This may be a common situation, robots clashing together and thus creating a compact black blob, originating a big obstacle if they are sufficiently lined up (Fig. 18a and b).

Although the computations for obstacle identification were in use during RoboCup 2009, their results are yet to be considered in the team strategy. Currently, obstacles are always considered unfriendly and thus to be avoided. Due to this fact, there is currently no data of in-game results for this part of the work.

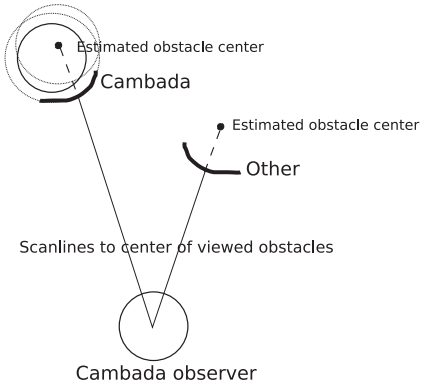

Fig. 16. When a CAMBADA robot is on, the estimated centers of the detected obstacles are compared with the known position of the team mates and tested; the left obstacle is within the CAMBADA acceptance radius, the right one is not.

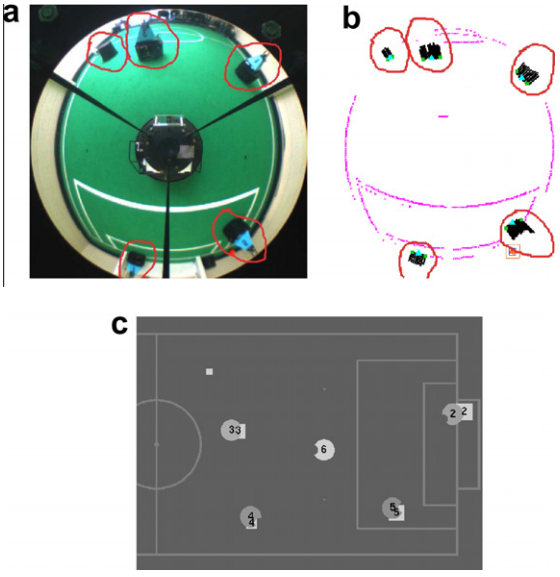

Fig. 17. Illustration of single obstacles identification. (a) Image acquired from the robot camera (obstacles for identification are marked). (b) The same image after processing. (c) Image of the control station. Each robot represents itself and robot 6 (the lighter gray) draws all the five obstacles evaluated (squares with the same gray scale as itself). All team mates were correctly identified (marked by its corresponding number over the obstacle square) and the opponent is also represented with no number.

Several captures of the obstacle identification algorithm described earlier were performed and analyzed, to further illustrate the effectiveness of the algorithm. The laboratory used for the tests receives natural light which can affect the vision processing algorithms. The presented results are not treated in any way to diminish the effects of natural light, as we are interested in understanding if the algorithms can cope with those conditions which can be found in real situations.

In the first test situation, a robot was positioned on the field at (−0.05, 1.88) while broadcasting its position. This robot will be referred to as *pivot*. Another robot was moving on a rectangular path around the pivot, and a capture of its data was done. This robot will be referred to as *observer*. This scenario is intended to give some insight about the performance of the identification when the team mates are static or nearly static (as is the case of set plays during the games. In these situations it is important to analyze passing lines). Fig. 19 is a graphic representation of the acquired data, with the pivot represented in black. The blue dots are the positions of the path taken by the observer, which covers the rectangular path

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

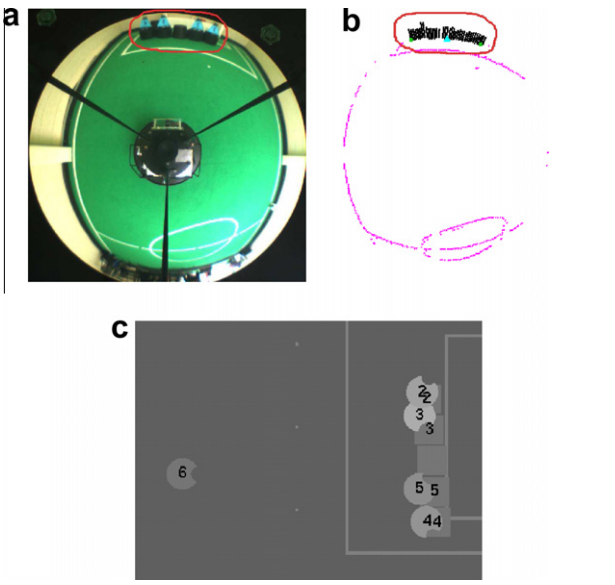

Fig. 18. Illustration of multiple obstacles identification. (a) Image acquired from the robot camera (obstacle for identification marked). (b) The same image after processing. Visually, the aligned robots are only one large obstacle. (c) Image of the control station. Each robot represents itself and robot 6 (the darker gray) draws all the five obstacles (squares with the same gray scale as itself). The visual obstacle was successfully separated into the several composing obstacles, and all of them were correctly identified as the correspondent team mate (marked by its corresponding number over the obstacle square) and the opponent is also represented with no number.

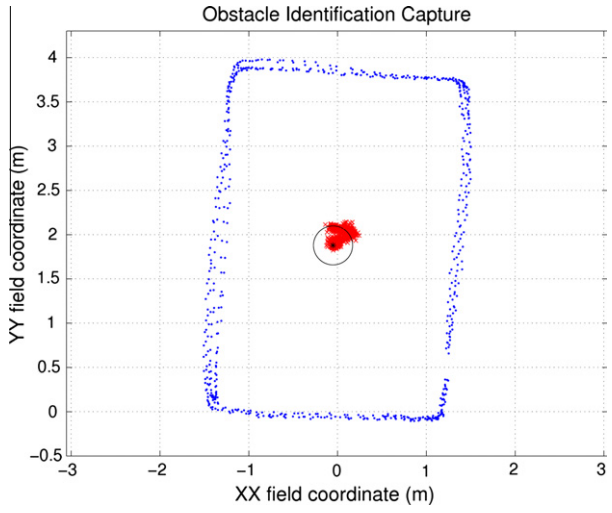

Fig. 19. Representation of a capture of the obstacle identification algorithm results. The path taken by the observer is represented by blue dots in the rectangular path taken. Near the center, the pivot shared position is represented by the black star and its limits by the black circle. The blob of red is the overlapping positions of the identified obstacle center, represented by a red cross. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

for three times. In each cycle, the center of the obstacle perceived by the observer is represented by a red '×'.

It is visible that, as expected, the obstacle position perceived by the observer is not exactly the pivot position. The capture in question is composed of 677 cycles. The identification of the obstacle as the correspondent team mate failed to succeed in only one cycle, which corresponds to a 99.85% success rate.

Considering that the pivot has 22 cm radius (although it is slightly bigger), the mean of the centers of the perceived obstacle is within the real area occupied by the pivot, at nearly 16 cm with a standard deviation of 10 cm (Table 1).

Another test scenario was considered for evaluation of the algorithm performance for moving obstacles. Several captures were performed to evaluate the performance of the algorithm when identifying a moving team mate. This set of six captures consisted on a robot observing a team mate moving around and registering the data about the obstacles. The path taken by the moving team mate is represented in Fig. 20. The number of failed identifications was greater when the moving robot was farther from the observer, as expected due to the noisy nature of the measurements.

The captures were performed throughout the day, with different lighting conditions but with the same robot calibration. Table 2 summarizes this set of captures, which revealed a total mean identification ratio of approximately 71%.

Table 1
The mean and standard deviation of the capture perceived obstacle position.

	Perceived obstacle	
	X	*Y*
Mean	0.05	2.01
Std	0.08	0.07

|Real − perceived| = 0.16.
|Std| = 0.10.

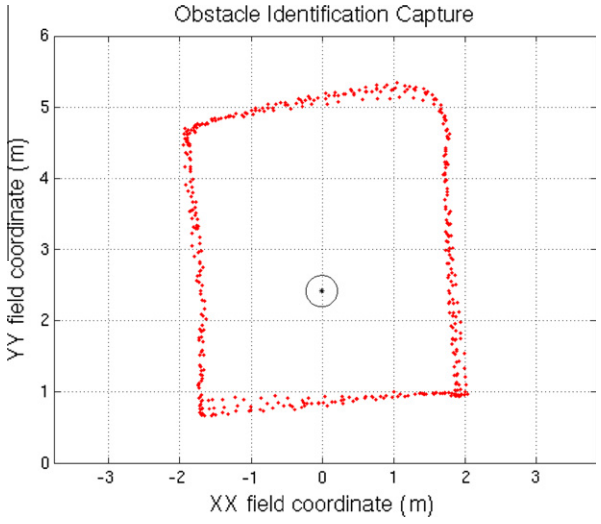

Fig. 20. Representation of the path taken by the team mate to identify (the red dots represent each communicated position). The observer position is represented by the black star and its limits by the black circle. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 2
The individual ratio of successful identification of the moving team mate for the several captures performed.

	Total cycles	Successes	%
Capture 1	1798	1319	73
Capture 2	1065	748	70
Capture 3	1528	1332	87
Capture 4	1162	769	66
Capture 5	1935	1278	66
Capture 6	2152	1411	66

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

5.3. Obstacle sharing

With the purpose of improving the global perception of the team robots, the sharing of locally known information is an important feature. Obstacle sharing allows the team robots to have a more global perception of the field occupancy, allowing them to estimate, for instance, passing and dribbling corridors more effectively.

However, one has to keep in mind that, mainly due to illumination conditions and eventual reflective materials, some of the detected obstacles may not be exactly robots, but dark shadowy areas. If that is the case, the simple sharing of obstacles would propagate an eventually false obstacle among the team. Thus the algorithm for sharing the obstacles makes a fusion of the several team mates information.

The fusion of the information is done mate by mate. After building the worldstate by its own means, the agent checks all the available obstacle information provided by team mates, one by one. Their obstacles are matched with the own ones. If the agent does not know an obstacle shared by the team mate, it keeps it in a temporary list of unconfirmed obstacles. This is done to all the team mates obstacles. When another team mate shares a common obstacle, that same obstacle is confirmed and is transferred to the local list of obstacles. In the current cycle, the temporary obstacles that were not confirmed are not considered. A robot does not use negative information from other robots to remove obstacles it actually saw from its local world model. An outline of the algorithm is presented next.

```
for c:=1 to total_number_of_team_mates
  for o:=1 to total_obstacles_of_team_mate
    for m:=1 to total_own_obstacles
      if m matches o
        I already know this obstacle, do nothing
      else
        if previously known by another team mate
          obstacle confirmed and added
        else
          obstacle considered temporarily
          waits for confirmation by another team
  mate
        endif
      endif
    endfor
  endfor
endfor
```

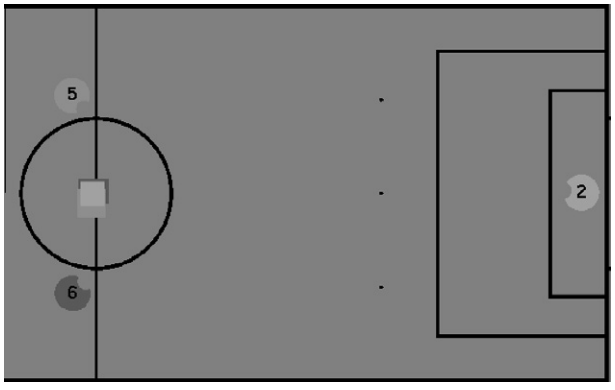

Fig. 21. Image of the control station showing an obstacle of robot 2 that was not seen by itself (on the center of the field). In this case it assumes the obstacle by confirmation of both robots 5 and 6.

The matching of the team mate obstacles with the own obstacles is done in a way similar to the matching of the obstacle identification with the team mate position described earlier. The CAMBADA team mate position in Fig. 16 is replaced by the current team mate obstacle for the matching test.

Fig. 21 shows a situation where robot 2, in the goal area was too far to see the obstacle on the middle of the field. Thus, it considered the obstacle in question, only because it is identified by both robots 5 and 6, as visible in the figure.

6. Conclusion and future work

The techniques chosen for information and sensor fusion proved to be effective in accomplishing their objectives. The Kalman filter allows to filter the noise on the ball position and provides an important prediction feature which allows fast detection of deviations of the ball path. The linear regression used to estimate the velocity is also effective, and combined with the deviation detection based on the Kalman filter prediction error, provides a faster way to recalculate the velocity in the new trajectory.

The improvement on obstacle treatment allows modifications on the overall team strategy, particularly regarding passing possibilities. It also allows the improvement of the robots movement, since team mate obstacles can have a different treatment than the opponents, because team mates have velocities and other information available.

The CAMBADA team obtained the 1st place in the last years of the Portuguese robotics open (Robótica 2007, Robótica 2008, Robótica 2009 and Robótica 2010), and internationally achieved 5th place in RoboCup 2007, 1st place in RoboCup 2008, 3rd place in RoboCup 2009 and 2nd place in GermanOpen 2010.

Although the described work proved to be effective and helped to achieve good results, improving is always the aim for this kind of project. Thus, improvements on the localization algorithm are desired, as well as a different way to disambiguate symmetric positions to eventually complement or replace the compass.

Another path to follow would be the improving of team strategy based on obstacle identification, creating new forms of cooperation and set plays for in-game situations.

Acknowledgment

This work was partially supported by project ACORD Adaptive Coordination of Robotic Teams, FCT/PTDC/EIA/70695/2006.

References

[1] Kitano H, Asada M, Kuniyoshi Y, Noda I, Osawa E. RoboCup: the robot world cup initiative. In: Proceedings of the first international conference on autonomous agents. New York (NY, USA): ACM; 1997. p. 340–7.
[2] MSL Technical Committee 1997–2009. Middle size robot league rules and regulations for 2009; 2008.
[3] Elmenreich W. Sensor fusion in time-triggered systems. Ph.D. thesis. Vienna (Austria): Technische Universitat Wien, Institut fur Technische Informatik; 2002.
[4] Metropolis N, Ulam S. The Monte Carlo method. J Am Stat Assoc 1949;44(247):335–41.
[5] Kalman R. A new approach to linear filtering and prediction problems. J Basic Eng 1960;82(1):35–45.
[6] Wan E, Merwe RVD. The unscented Kalman filter for nonlinear estimation. In: IEEE adaptive systems for signal processing, communications, and control symposium; 2000. p. 153–8.
[7] Luo R, Yih C, Su K. Multisensor fusion integration: approaches, applications, and future research directions. IEEE Sens J 2002;2(2):107–19.
[8] Leonard J, Durrant-Whyte H. Mobile robot localization by tracking geometric beacons. IEEE Trans Robotics Autom 1991;7(3):376–82.
[9] Dellaert F, Fox D, Burgard W, Thrun S. Monte Carlo localization for mobile robots. In: IEEE international conference on robotics and automation; 1999. p. 1322–8.

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011

[10] Fox D, Burgard W, Thrun S. Markov localization for mobile robots in dynamic environments. J Artif Intell Res 1999;11:391–427.
[11] Mourikis A, Roumeliotis S. Performance analysis of multirobot cooperative localization. IEEE Trans Robotics 2006;22(4):666–81.
[12] Durrant-Whyte H, Henderson T. Multisensor data fusion. In: Siciliano B, Khatib O, editors. Springer handbook of robotics. Springer; 2008.
[13] Bejczy A, Dias J. Editorial: integration of visual and inertial sensors. J Robotic Syst 2004;21(2):41–2.
[14] Alenyá G, Martínez E, Torras C. Fusing visual and inertial sensing to recover robot ego-motion. J Robotic Syst 2004;21(1):23–32.
[15] Chroust S, Vincze M. Fusion of vision and inertial data for motion and structure estimation. J Robotic Syst 2004;21(2):73–83.
[16] Thrun WBS, Fox D. Probabilistic robotics. The MIT Press; 2005.
[17] Siciliano B, Khatib O. Springer handbook of robotics. Springer; 2008.
[18] Lauer M, Lange S, Riedmiller M. Modeling moving objects in a dynamically changing robot application. In: Furbach U, editor. KI 2005: advances in artificial intelligence. Lecture notes in computer science, vol. 3698. Springer; 2005. p. 291–303.
[19] Xu Y, Jiang C, Tan Y. SEU-3D 2006 soccer simulation team description. In: CD proc of RoboCup symposium 2006, Bremen, Germany; 2006.
[20] Marcelino P, Nunes P, Lima P, Ribeiro MI. Improving object localization through sensor fusion applied to soccer robots. In: Proc scientific meeting of the portuguese robotics open – Robótica 2003, Lisbon, Portugal; 2003.
[21] Ferrein A, Hermanns L, Lakemeyer G. Comparing sensor fusion techniques for ball position estimation. In: Bredenfeld A, Jacoff A, Noda I, Takahashi Y, editors. RoboCup 2005: robot soccer world cup IX. Lecture notes in computer science, vol. 4020. Springer; 2006. p. 154–65.
[22] Lauer M, Lange S, Riedmiller M. Calculating the perfect match: an efficient and accurate approach for robot self-localization. In: Bredenfeld A, Jacoff A, Noda I, Takahashi Y, editors. RoboCup 2005: robot soccer world cup IX. Lecture notes in computer science, vol. 4020. Springer; 2006. p. 142–53.
[23] Neves A, Martins D, Pinho A. A hybrid vision system for soccer robots using radial search lines. In: Lopes LS, Silva F, Santos V, editors. Proc of the 8th conference on autonomous robot systems and competitions, Portuguese robotics open – Robótica 2008, Aveiro, Portugal; 2008. p. 51–5.
[24] Bishop G, Welch G. An introduction to the Kalman filter. In: Proc of SIGGRAPH, Course 8, No. NC 27599-3175. NC (USA): Chapel Hill; 2001.
[25] Motulsky H, Christopoulos A. Fitting models to biological data using linear and nonlinear regression. GraphPad Software Inc.; 2003.
[26] Neves A, Corrente G, Pinho A. An omnidirectional vision system for soccer robots. In: Neves J, Santos MF, Machado JM, editors. Progress in artificial intelligence. Lecture notes in artificial intelligence, vol. 4874. Springer; 2007. p. 499–507.
[27] Cunha B, Azevedo J, Lau N, Almeida L. Obtaining the inverse distance map from a non-SVP hyperbolic catadioptric robotic vision system. In: Visser U, Ribeiro F, Ohashi T, Dellaert F, editors. RoboCup 2007: robot soccer world cup XI. Lecture notes in artificial intelligence, vol. 5001. Springer; 2008. p. 417–24.

Please cite this article in press as: Silva J et al. World modeling on an MSL robotic soccer team. Mechatronics (2010), doi:10.1016/j.mechatronics.2010.05.011