20th RoboCup International Symposium, Leipzig, Germany, July 2016.

Cooperative Sensing for 3D Ball Positioning in the RoboCup Middle Size League

Wouter Kuijpers[1], António J.R. Neves[2], and René van de Molengraft[1]

[1] Departement of Mechanical Engineering, Eindhoven University of Technology, The Netherlands, w.j.p.kuijpers@student.tue.nl, m.j.g.v.d.Molengraft@tue.nl
[2] IRIS Lab/IEETA/DETI, University of Aveiro, Portugal, an@ua.pt

Abstract. As soccer in the RoboCup Middle Size League (MSL) starts resembling human soccer more and more, the time the ball is airborne increases. Robots equipped with a single catadioptric vision system will generally not be able to accurately observe depth due to limited resolution. Most teams, therefore, resort to projecting the ball on the field. Within the MSL several methods have already been explored to determine the 3D ball position, e.g., adding a high-resolution perspective camera or adding a Kinect sensor. This paper presents a new method which combines the omnivision camera data from multiple robots through triangulation. Three main challenges have been identified in designing this method: *Inaccurate projections*, *Communication delay* and *Limited amount of data*. An algorithm, considering these main challenges, has been implemented and tested. Performance tests with a non-moving ball (static situation) and two robots show an accuracy of 0.13 m for airborne balls. A dynamic test shows that a ball kicked by a robot could be tracked from the moment of the kick, if enough measurements have been received from two peer robots before the ball exceeds the height of the robots.

1 Introduction

The Robot Soccer World Cup (RoboCup) Federation is an international organization which focuses on the promotion of robotics and Artificial Intelligence (AI) research, by offering a publicly appealing challenge: build robots that play soccer[3]. In the Soccer Middle Size League (MSL), robots of no more than 50 cm in diameter and 80 cm in height, play soccer in teams of five. At the moment of writing, 26 teams from all over the world compete in this league.

During a game of soccer, the position and velocity of the ball are of great importance. To detect the position of the ball, most teams have equipped their robots with a catadioptric vision system, which also serves a number of other purposes. Although this is not prescribed by the league, it has been widely adopted because of its price versus value as sensor. As soccer in the MSL starts resembling human soccer more and more, the time the ball is airborne increases.

[3] RoboCup Homepage: http://www.robocup.org/

Robots equipped with only a catadioptric vision system have a single camera, hence they will, generally, not be able to accurately observe depth due to limited resolution. Most teams, therefore, resort to projecting the ball on the field.

Fig. 1 shows that projecting the ball (x_b, y_b, z_b) on the field leads to a false projection. The ball will be detected in position (x_p, y_p), where $x_p \neq x_b$ and $y_p \neq y_b$ when the ball is airborne. The positioning of the robot will benefit when the correct ball position (x_b, y_b), instead of the false projection is being used. With the height of the ball (z_b) it is possible to calculate where the ball bounces, at these locations the ball could be intercepted after a lob pass.

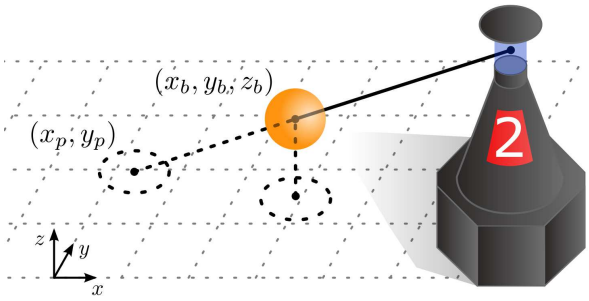

Fig. 1: A MSL robot detecting an airborne ball (x_b, y_b, z_b), creating a false projection (x_p, y_p).

This paper presents the design, implementation and testing of a new algorithm based on cooperative sensing and triangulation. Section 2 presents related research in the area of cooperative sensing and distributed sensor fusion. Section 3 presents the main challenges faced during the design and implementation of a multi-robot triangulation algorithm. Section 4 follows up on that by presenting the general structure of the algorithm. In this section special attention will be paid to how the previously defined challenges have been considered. Section 5 presents the results obtained during tests with the algorithm, results with a non-moving (static) and moving (dynamic) ball are presented. Section 6 concludes this paper with concluding remarks and recommendations for future work in this area.

2 Related Work

The topic of multi-robot tracking is one studied in many different applications and different communities as presented by [4]. To structure this broad field, [4] presents a unifying taxonomy to classify the various missions related to the topic. This paper presents a *Target Localization*-application. The application discussed

here is characterized by (mostly) homogeneous teams of robots.[4] Target localization is also possible with heterogeneous teams as shown in [2].

An application using multiple sensors is presented in [7]. The part of the research presented in [7] focusing on stereo vision with catadioptric cameras is similar to the method presented here; triangulation. In this research the sensors are not connected to the same platform (robot), this fact makes the implementation of the triangulation algorithm more difficult because of data communication delay. This also shows from the research in [8], which researches cooperative map building in the context of autonomous vehicles. The vehicles are communicating via an IEEE 802.11n wireless interface, the data communication delay is taken into consideration by calculating and applying the coordinate offset. Although this method provides a solution in the context of autonomous vehicles, its results will deteriorate in an environment as dynamic as the MSL.

Within the MSL itself a substantial amount of research has been performed in the field of 3D ball detection and tracking. In [9] a solution is presented where aerial balls are detected using a front-facing perspective camera, the distance from the camera to the ball along the focal axis is calculated using the number of pixels occupied by the ball in the image. In [11] the robot is equipped with an additional front facing perspective camera, the measurement is combined with omnivision using triangulation. Instead of adding a perspective camera to the robot, [3] and [6] present a method using an additional Kinect sensor[5]. This paper presents a method in which no additional sensors have to be added and which is compatible with the "league standard" omnivision system.

3 Main Challenges

Before the design of the triangulation algorithm, a set of main challenges has been identified. Basically, the triangulation algorithm calculates the intersection of two lines, defined by the position of the robot (and its height) and the projection of the respective robot. The main challenges presented in this section, hinder the implementation of this 'simple' algorithm. The main challenges are: *Inaccurate projections*, *Communication delay* and *Limited amount of data*. In this section each challenge will be treated separately, elaborating on where it originates from and how it affects the triangulation of omnivision camera data.

3.1 Inaccurate Projections

The vision software is responsible for detecting the ball in the image captured by the camera in the catadioptric vision system. The exact structure of this

[4] some teams have equipped the goalkeeper with different or additional sensors. [1]
[5] Kinect: `https://dev.windows.com/en-us/kinect`

software differs between teams, but frequently the main structure is comparable. The image captured is segmented based on colors, and from this blobs are identified. The blobs are ranked according to the probability of the blob being the ball, based on: the color of the ball, its size and shape. The position of the center of the blob in the image captured, is communicated to the rest of the system as the projection [10]. The projection might be inaccurate because of the limited resolution of the catadioptric vision system or because of motion blur e.g.

The 'simple' algorithm presented at the start of this section, could rely on the calculation of the intersection of two lines. But due to the inaccuracies in projection, or in the localization of the robot, the lines of sight might not have an intersection. An algorithm searching for the intersection of the lines will therefore not suffice in this context.

3.2 Communication Delay

To be able to triangulate lines of sight, a robot requires the position and projection of at least one other robot, which means that the robots have to communicate information. Sharing information between robots induces additional delay on the information. To communicate, several teams in the MSL use the Real-time Data Base (RTDB) [5], where robots communicate via a distributed shared memory and an adaptive TDMA protocol for wireless communication.

If the effects of this additional delay would not be considered in the triangulation algorithm, lines of sight from different time instants might be triangulated. During a MSL game the ball can reach speeds of up to 11 m/s, with a communication delay of 20 ms, triangulating a different time instant introduces an error of 0.22 m into the triangulation algorithm. The effect of this error on the triangulated ball, depends on the positions of the robots relative to the ball and the height of the ball.

3.3 Limited Amount of Data

Aside from the delay on the data received from peers, the data from peers might also not be available at every time instant. If a peer, is not able to detect the ball because it is outside the field of view or the line of sight of the peer is obstructed by another robot, the ball is not detected by the peer and therefore not communicated. Secondly, the robot-robot communication will often not run at the same frequency as the acquisition of images by the vision system.

The combination of the two factors above might lead to time instants where no information or information from only one robot is available. It is desirable that the triangulation algorithm provides an output also at these time instants.

4 Triangulation Algorithm

In this section we propose a new triangulation algorithm, the emphasis here will be on how the main challenges, presented in the previous section, are considered in the design of the algorithm.

Fig. 2 shows an example of a state of the triangulation algorithm. In this example, the current time is t_n and the algorithm represented executes on Robot 1. Robot 1 therefore has all position and projection information (represented by the circles) from itself, up to t_n. The information from the two other robots still has to arrive. Because of the communication delay, the data from Robot 2 is available up to t_{n-3} and data from Robot 3 is available up to t_{n-2}. The proposed algorithm triangulates the most recent time instant where information from the most robots is available; in this case t_{n-4}. The algorithm applies triangulation to pairs of lines of sight. If more than two lines of sight are available (in the case of t_{n-4}) the lines of sight are triangulated pairwise, the results of these pairwise triangulations are combined by averaging. Pairwise triangulation has been chosen because the method for finding the minimum distance between two lines is one which is relatively easy and therefore more suitable to be implemented in a real-time system, compared to the method used for triangulating n lines. After triangulation, the 3D ball position is filtered by means of a Kalman filter. This results in a filtered 3D ball state at t_{n-4}. The state at t_{n-4}, in combination with the ball model, is used to determine the 3D ball state at t_n.

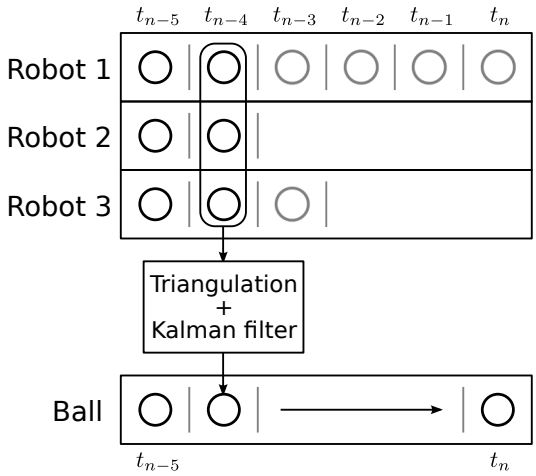

Fig. 2: An example of a state of the triangulation algorithm, used by three robots. The circles in the rows denoted by "Robot" represent communicated data, circles in the row denoted by "Ball" denote data regarding the state of the ball.

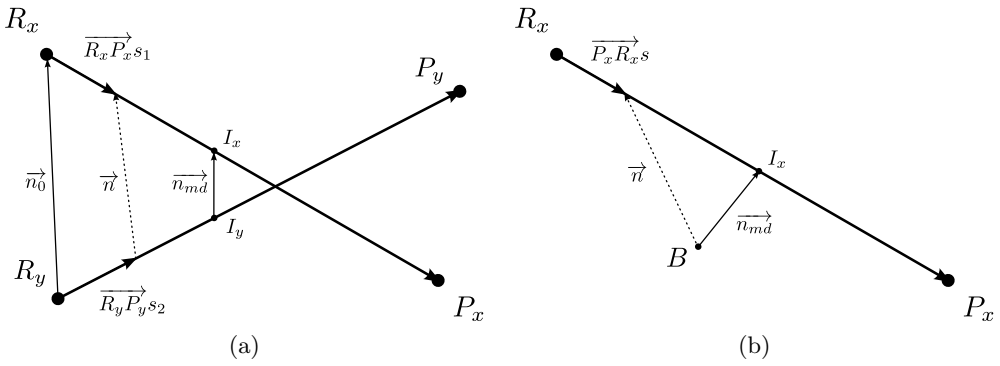

Fig. 3: Graphical representation of a minimum distance algorithm applied to: (a) two lines of sight (b) a line of sight and the predicted position of the Kalman filter.

4.1 Inaccurate Projections

Due to inaccuracies in the projections, the lines of sight of the robots might not cross. Therefore, a minimum distance algorithm is in place. Fig. 3(a) graphically represents the minimum distance algorithm. The lines $\overrightarrow{R_xP_x}$ and $\overrightarrow{R_yP_y}$ represent, respectively, the lines of sight of robot x and y. To find the minimum distance $\overrightarrow{n_{md}}$, the line $\overrightarrow{n}$ between two arbitrary points, one on each line, is parameterized

$$\overrightarrow{n} = \overrightarrow{n_0} + \overrightarrow{R_xP_x}s_1 - \overrightarrow{R_yP_y}s_2. \tag{1}$$

the minimum distance line $\overrightarrow{n_{md}}$ is perpendicular to both lines of sight, hence

$$\begin{cases} \overrightarrow{n} \cdot \overrightarrow{R_xP_x} = 0 \\ \overrightarrow{n} \cdot \overrightarrow{R_yP_y} = 0 \end{cases}. \tag{2}$$

Solving this system of equations yields a closed-form solution for the parameters s_1 and s_2, which are used to describe the intersection points I_x and I_y. The 3D ball position communicated to the rest of the system is in between I_x and I_y. This latter average can be replaced by a weighted average based on the length of e.g. $\overrightarrow{R_xP_x}$. The larger the distance between the robot and projection, the less accurate it will be.

4.2 Communication Delay

Due to the communication delay in the robot-robot communication, the arrival of information from peers is delayed. The proposed triangulation algorithm considers this effect by storing the information from the robots in a data buffer. The

data buffer is graphically represented in the top of Fig. 2.

Triangulation on data from time instants in the past requires the storage of this information. The data buffer stores the information from the peer robots and the robot itself in this buffer, quantized to time instants defined by the execution times of the robot.

4.3 Limited Amount of Data

The data buffer could contain: no information or information from only a single robot. To find the 3D ball position, the 'simple' algorithm would require at least two lines of sight at a certain time instant.

In case no information from the robot itself and from its peers is available, the algorithm applies the model to the previous state to estimate the ball position at the current time instant. If the ball experiences disturbances: e.g. bouncing off other robots or being kicked by another robot this method will show serious deviations as these disturbances are not modeled.

In case only one robot detects the ball, it is not possible to obtain a 3D ball position using the 'simple' algorithm presented previously. It is, however, desirable to include the new information in the derivation of the 3D ball position at the current time instant. The algorithm therefore implements a minimum distance algorithm similar to that presented before, see Fig. 3(b). In the case of only one line of sight, the minimum distance $\overrightarrow{n_{md}}$ between the line of sight $\overrightarrow{R_x P_x}$ and the predicted position of the Kalman filter B for that particular time instant will be determined. $\overrightarrow{n_{md}}$ is in that case defined as

$$\overrightarrow{n_{md}} = \min_s \sqrt{\overrightarrow{n} \cdot \overrightarrow{n}} = \min_s \overrightarrow{n} \cdot \overrightarrow{n}. \tag{3}$$

Equating $\frac{d(\overrightarrow{n} \cdot \overrightarrow{n})}{ds}$ to zero yields the value of s which parameterizes line $\overrightarrow{n_{md}}$. The 3D ball position communicated to the rest of the system is in between I_x and B. This latter average can be replaced by a weighted average based on e.g. the length of $\overrightarrow{R_x P_x}$. Note that this method assumes that the predicted position of the Kalman filter B is an accurate representation for the ball position, so the ball should not be affected by disturbances during the time, the prediction is made over.

5 Results

In this section a performance analysis of the algorithm will be presented, it consists out of two tests: static and dynamic. This section is structured accordingly.

5.1 Static Test

To validate the implementation of the algorithm and to quantify the accuracy of the algorithm in a static environment, a set of static tests is defined. The ball is positioned on 5 predefined (x, y) positions[6] which make up set $\mathcal{P}$:

$$\mathcal{P} = \{(0, 6.05), (0, 3.03), (0, 1.94), (1.5, 3.03), (-1.5, 3.03)\}. \tag{4}$$

For each position in $\mathcal{P}$, two different heights are used: the ball was placed on the field $z = 0.11\ m$, and on a green box[7] $z = 0.42\ m$. For each of the ball position (x, y, z), 200 measurements are logged and the mean error and standard deviations in the 200 measurements are determined. A photo during execution of $\mathcal{P}_3$ with $z = 0.42\ m$, is presented in Fig. 4.

For each of the ball positions (x, y, z) the mean error and standard deviation within the measurement set is determined, the results are shown in Table 1. These results are also compiled into Fig. 5. In this figure the robot positions are denoted by a $\times$. The mean detected position is shown by the red and yellow dots, for each of the (ground truth) positions on the field; represented by the black dots. For the figure the detected ball positions have been quantized to two values of z: $z = 0.11\ m$ and $z = 0.42\ m$, presented in different plots.

The analysis of the mean error table, shows that the mean error when the ball is lifted ($z = 0.42\ m$) is always higher than when it is on the field ($z = 0.11\ m$). The increase in (mean) error was probably caused by the less accurate detection of the ball. The ball will appear closer to the edge of the omnivision image, where the pixel density is less. The inaccuracy caused by the latter, combined with the other sources of inaccuracies mentioned in subsection 3.1 is also enlarged as the distance over which the ball is projected increases.

Evaluating the performance of the triangulation algorithm in the static situation shows a mean error of $0.13\ m$ when the ball is airborne and $0.08\ m$ when it is on the field. The standard deviation on the measurements is always lower than $2\ cm$. This performance is considered to be satisfactory for use in the MSL.

5.2 Dynamic Test

To see if the triangulation algorithm is suitable for real-time 3D ball positioning by robots in the MSL, the algorithm has to be tested under dynamic game situations. The kick from a soccer robot has been selected as a dynamic event. The challenge that comes with tracking a kick, is the ball leaving the detectable space

[6] with respect to frame presented in: `http://wiki.robocup.org/images/1/1a/MSL_WMDataStruct.pdf`

[7] a green box has been selected, to ensure that the actual detection of the ball by the vision module, is not affected by the presence of the box.

Fig. 4: Photo taken during the execution of $\mathcal{P}_3$ with $z = 0.42\ m$, from the static tests.

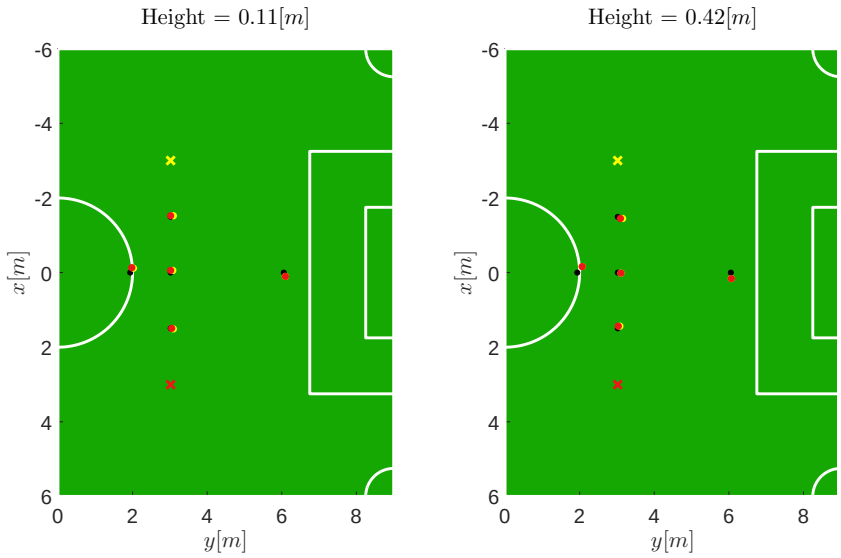

Fig. 5: Graphical representation of the results of the static test. The positions in (ground truth) set $\mathcal{P}$ are represented by black dots. The mean detected positions are represented by the red and yellow dots; corresponding to the robot, represented by the respective colored cross.

of the robots; the ball is not being detected when it exceeds the robot height. This means that before the ball leaves the detectable space of the robot, the triangulation algorithm has to have received enough measurements from peer-robots for its Kalman filter to have estimated the state accurately enough. If the state is estimated accurately, the point where it reenters the observable space can be calculated with decent accuracy. This subsection provides an analytic analysis of the dynamic tests. A photo of the setup is presented in Fig. 6.

Table 1: The mean error (μ) and standard deviation (σ) for both robots in the static test. $\uparrow$ represents $z = 0.42\ m$ and $\downarrow$ represents $z = 0.11\ m$. The averages for all position are presented in the last row, denoted by $\mathbb{E}$.

		Robot 1		Robot 2	
		$\mu\ [m]$	$\sigma\ [cm]$	$\mu\ [m]$	$\sigma\ [cm]$
$\mathcal{P}_1$	$\uparrow$	0.159	0.45	0.159	0.43
	$\downarrow$	0.117	0.51	0.106	0.55
$\mathcal{P}_2$	$\uparrow$	0.083	0.52	0.091	0.70
	$\downarrow$	0.076	0.92	0.067	0.84
$\mathcal{P}_3$	$\uparrow$	0.208	0.34	0.205	0.32
	$\downarrow$	0.142	0.36	0.138	1.01
$\mathcal{P}_4$	$\uparrow$	0.087	1.96	0.065	0.57
	$\downarrow$	0.073	1.38	0.023	1.32
$\mathcal{P}_5$	$\uparrow$	0.139	1.02	0.090	1.17
	$\downarrow$	0.070	1.96	0.034	0.89
$\mathbb{E}$	$\uparrow$	0.135	0.75	0.121	0.64
	$\downarrow$	0.095	1.03	0.074	0.92

The detected 3D ball positions from the robots without the ball (see Fig. 6) are presented in Fig. 7. The points in the figure represent the (filtered) triangulated balls, by the respective robots. The points are connected by lines to emphasize the sequence of points. Fig. 7 shows that 8 samples are obtained from the moment the ball is kicked to the moment the ball leaves the observable space. The robots are positioned (x, y) at $(0, 2)$, $(0, 4)$ and $(-2, 3)$[8], where the latter robot is going to kick the ball.

The initial state estimate x_0 represents a ball lying at the midpoint of the field. At the start of the experiment the Kalman filter is given some time to converge to the state presented in Fig. 6. These tests have been executed with a scalar matrix $Q = 1 \cdot I$, $R = 0.1 \cdot I$ and $P_0 = 1 \cdot I$. These matrices are used as tuning parameters for the Kalman filter, the results presented in Fig. 7 have been achieved after a rough tuning. The tuning of the Kalman filter in this case allows the Kalman filter to quickly converge to the actual ball trajectory after the kicking event, which was not included in the model. This, however, makes the

[8] with respect to frame presented in: `http://wiki.robocup.org/images/1/1a/MSL_WMDataStruct.pdf`

estimation of the Kalman filter more susceptible to measurement inaccuracies, which can clearly be seen from the inaccuracies in the triangulated ball output.

Fig. 6: Photo of the setup of the dynamic test.

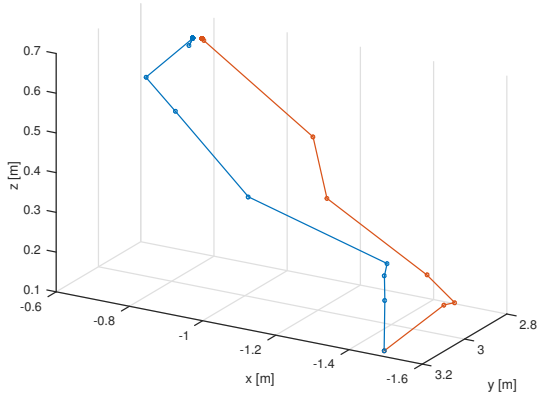

Fig. 7: The results of the dynamic test, where the ball trajectory, as detected by two robots (blue for the robot at $(0, 4)$ and orange for the robot at $(0, 2)$), is presented. The colored dots represent the detections of the ball.

6 Conclusion and Future Work

This paper presents a method for 3D ball positioning in which no additional sensors have to be added and which is compatible with the "league standard"

omnivision system. The algorithm was implemented on the robots of both team CAMBADA and team Tech United. Static tests show that the error increases when the ball is airborne. An increase of ≈ 4 *cm* is observed if the ball is at a height of 0.42 *cm* compared to the situation where the ball is on the field. Dynamic tests show that it is difficult to track a kick from a robot. The Kalman filter has to react on the kick of the robot within samples, but it should not be susceptible to measurement inaccuracies.

The implementation on both teams is now directed towards testing. Before it is suitable for competition, the algorithm has to be integrated in the software more closely. For example: the projection of the ball can appear outside the field, both teams ignore the projection in this case. Tech United also employs Kinect sensors for ball detection, sensor fusion is therefore a point of attention as well.

References

1. Cunha, B., Neves, A.J.R., Dias, P., Azevedo, J.L., Lau, N., Dias, R., Amaral, F., Pedrosa, E., Pereira, A., Silva, J., Cunha, J., Trifan, A.: CAMBADA '2015: Team Description Paper (2015),
2. Dias, A., Almeida, J., Silva, E., Lima, P.: Multi-robot cooperative stereo for outdoor scenarios. In: 2013 13th International Conference on Autonomous Robot Systems. pp. 1–6. IEEE (apr 2013),
3. Neves, A.J.R., Trifan, A., Dias, P., Azevedo, J.L.: Detection of Aerial Balls in Robotic Soccer Using a Mixture of Color and Depth Information. In: 2015 IEEE International Conference on Autonomous Robot Systems and Competitions. pp. 227–232. Aveiro (2015),
4. Robin, C., Lacroix, S.: Multi-robot target detection and tracking: taxonomy and survey. Autonomous Robots (August) (aug 2015),
5. Santos, F., Almeida, L., Lopes, L.S., Azevedo, J.L., Cunha, B.: Communicating among robots in the RoboCup Middle-Size League. In: RoboCup 2009: Robot Soccer World Cup XIII. pp. 320–331 (2010)
6. Schoenmakers, F., Koudijs, G., Lopez, C., Briegel, M., van Wesel, H., Groenen, J., Hendriks, O., Klooster, O., Soetens, R., Van De Molengraft, R.: Tech United Eindhoven Team Description 2013 - Middle Size League (2013),
7. Schönbein, M., Kitt, B., Lauer, M.: Environmental Perception for Intelligent Vehicles Using Catadioptric Stereo Vision Systems. ECMR pp. 189–194 (2011),
8. Seong-Woo Kim, Zhuang Jie Chong, Baoxing Qin, Xiaotong Shen, Zhuoqi Cheng, Wei Liu, Ang, M.H.: Cooperative perception for autonomous vehicle control on the road: Motivation and experimental results. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5059–5066. IEEE (nov 2013),
9. Silva, J., Antunes, M., Lau, N., Neves, A.J.R., Lopes, L.S.: Aerial ball perception based on the use of a single perspective camera. In: Lecture Notes in Computer Science. vol. 8154 LNAI, pp. 235–246. Aveiro (2013)
10. Trifan, A., Neves, A.J.R., Cunha, B., Azevedo, J.L.: UAVision : A modular time-constrained vision library for soccer robots. RoboCup 2014: Robot World Cup XVIII 8992, 490–501 (2015),
11. Voigtländer, A., Lange, S., Lauer, M., Riedmiller, M.: Real-time 3D ball recognition using perspective and catadioptric cameras. Ecmr 2007 pp. 1–6 (2007),