Unknown-Color Spherical Object Detection and Tracking

Anže Troppan, Eduardo Guerreiro, Francesco Celiberti, Gonçalo Santos, Aamir Ahmad and Pedro U. Lima

Institute for Systems and Robotics, Instituto Superior Técnico, Lisboa, Portugal

***Abstract*—Detection and tracking of an unknown-color spherical object in a partially-known environment using a robot with a single camera is the core problem addressed in this article. A novel color detection mechanism, which exploits the geometrical properties of the spherical object's projection onto the image plane, precedes the object's detection process. A Kalman filter-based tracker uses the object detection in its update step and tracks the spherical object. Real robot experimental evaluation of the proposed method is presented on soccer robots detecting and tracking an unknown-color ball.**

I. Introduction

Detecting and tracking relevant objects in a known or unknown environment forms a vast area of research not only in the domain of image processing and computer vision [1] [2] [3] [4], but also in that of mobile robotics [5] [6] [7] [8]. Often the objective in the former is to classify complex objects based on their color and shape irregularities [1]. However, in mobile robotics the focus is more on computationally fast methods for object tracking (including detection) and thereby using the objects' information to achieve other complex goals, e.g, robot's self-localization [7] and high-level decision making [9]. The gain in an object tracker's execution speed is due to using the object's shape as a prior knowledge and/or using less-complex shaped objects, e.g, spheres. Although it is often practical to assume the presence of less-complex shaped objects in most environments, the same cannot be true with regard to the object's color. Moreover, the lighting conditions of the environment irregularly change the apparent colors of the objects over time rendering the prior knowledge of the object's color useless.

In this article we present a method to solve the problem of detecting and tracking spherical-shaped objects in a partially-known environment without the necessity of the object's color information as a prior knowledge of the tracker. The partially-known environment implies that the most dominant colors contributing to the environment's background are known beforehand. The method essentially consists of the following three modules:

- automatic color detection of a spherical-shaped object,
- spherical object detection method and
- a Kalman filter-based (KF) tracker that uses the spherical object detection method in its update step.

The automatic color detection, which consists of a sweep over the HSV color space, is performed only once before initiating the tracker. Therefore, the object should be in the field of view of the camera at the beginning of the tracking process. The detected color is subsequently used by the spherical object detection method which along with the KF-based tracker runs continuously to track the object. Since the automatic color detection method itself is computationally fast, it can be scheduled to run intermittently without the need of manual intervention in environments where the lighting changes frequently causing the apparent color of the object to change. In order to experimentally validate our proposed method, the tracker was implemented on real soccer robots to track an unknown-color soccer ball.

The rest of the article is organized as follows. In Section II we briefly overview the existing literature in the context of object detection and tracking. The novel contributions of the article concerning automatic color detection is presented in Section III along with the spherical object detection and the KF-based tracker's theoretical details. Real robot experimental results are presented in Section IV, followed by Section V where we conclude the article with comments on future work.

II. Related Work

An extensive literature exists in the area of object detection where researchers have explored concepts ranging from contour recognition, e.g., Hough transform (HT) [10] to structure tensor techniques [4]. While some of these assume a prior knowledge of the object's color [8], innovative algorithms have been proposed recently to overcome this assumption, e.g, [5] [6]. In [5] the authors present a color histogram mismatch-based method to distinguish a spherical object of known size from the background. This removes the dependency on the prior color information and enables the spherical object detection in 3D space. However, since its accuracy depends on the number of pixels required to perform the histogram mismatch (the higher the number of pixels used, the better the accuracy is) at every frame, the execution speed is adversely affected causing the method to be inefficient for real-time applications.

In [11] the authors present a circular HT-based method to identify the circular projections of the spherical object to detect them. Apart from being computationally expensive, the method explicitly requires a fine-tuning of the edge-detection system, required by the HT, every time the lighting condition of the environment changes. A possible solution to this problem is proposed in [12], where the image projection of the spherical object is an exact ellipse and radial and rotary scan lines are used to look for matching color variations to detect the outer

978-989-97531-2-9

edge of the ellipses. The method is fast and suitable for real-time detection of unknown-color spherical objects. However, it is too specific to a certain kind of vision system that consists of a special mirror with a hyperbolic part in the middle, a horizontal isometric mirror and a vertical isometric mirror in the outer parts. In this article we show that our method is fast, adapts automatically to lighting changes in the environment and requires an easily available perspective lens-based camera making it diversely usable.

III. Detection and Tracking Mechanism

To detect and track a specific type of object in an image, some of its characteristics must be known beforehand. These are usually the object's shape and color. In this article we deal with spherical shaped objects, henceforth referred simply as objects. Detecting a spherical object assuming a prior knowledge of its color is a relatively easy and previously solved problem. However, in order to solve the generic problem of detecting objects in any lighting condition, the known-color assumption needs to be dropped. Our solution to this problem involves automatic color detection (ACD) of the object before initiating its detection and tracking process. This consists of sweeping the *hue-saturation-value* (HSV) color space in *hue* intervals, and then applying the color detection mechanism, explained further, to each *hue* interval. The output of this method is the most likely *hue* interval that corresponds to the color of the object. This process should be repeated after a pre-determined time interval to adapt to varying lighting conditions of the environment in which the detection is done. Since the ACD is a fast method (< 1 second) the tracker can easily afford to execute the ACD without affecting its own execution speed. The resulting *hue* interval information is then used by the object detection mechanism (ODM) by filtering each image frame using only that *hue* interval.

The approach mentioned above includes the use of only a perspective lens-based camera fixed to the robot and pitched down towards the ground plane (GP). The GP is where the robot moves (translates on the GP's $X-Y$ plane and rotates around an axis passing through the robot's center of mass and perpendicular to the ground plane). To calculate the distance to the object from the robot we use the perspective transformation between the the image frame and the GP, assuming that the robot is at the origin of the coordinates and the object is on the GP. This allows a linear pixel-to-meter relationship for every pixel belonging to the GP. Once the pixel corresponding to the object's contact point with the GP is known (by the ODM), we use the aforementioned relationship to calculate the distance to it in meters. Furthermore, using simple geometric calculations, the object's center's coordinates in the robot-centric frame of reference is calculated. Finally, the ODM acts as a classifier for the update step of a KF-based tracking method to track the object's position and velocity. The robot's odometry is used in the KF's prediction step.

As mentioned previously, we assume that the detection and tracking is done in a partially known environment meaning that the GP consists of a few dominant colors which contribute to a large, useless part of the image. Assuming the prior knowledge of this information, we simply eliminate their corresponding image pixels, facilitating a much faster ACD of the object and the ODM.

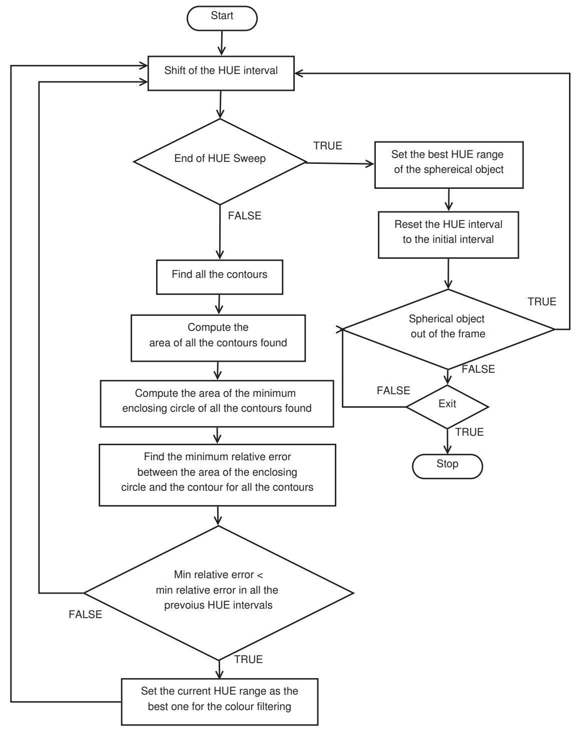

Fig. 1: Automatic color detection algorithm

A. Automatic Color Detection (ACD) Mechanism

The ACD operates only on the first image (also assuming that the object is present in that image) but can be executed later if required, as mentioned previously. At first the acquired image frame is blurred to remove noise. A color filter (consisting of the known dominant colors of the GP) is applied to this to remove the regions consisting of the GP background. This leaves only the object to be detected along with other objects which might be present on the GP. Note that a rather trivial assumption here is that the GP's dominant color is not the same as that of the object. Subsequently the following two steps are performed:

- **Sweep of the HSV color space in *hue* intervals:** Consecutive color filters are applied for every interval in the *hue* range. Although the *hue* space is swept in the tonality in intervals of ten, the *value* and *saturation* intervals are constant in order to get different intensities for each color. For each *hue* interval we do the following. Take the GP color-filtered image and re-filter it for the *hue* interval in consideration. The output image now consists only of the pixels belonging to

978-989-97531-2-9

that *hue* interval. We then find the contours on this output image. If the area of any contour is less than a certain predefined threshold value, it is discarded as noise. We then compute the minimum enclosing circles for each of the remaining contour and the area of those enclosing circles. The relative error between the area of each enclosing circle and the respective internal contour is also computed. The minimum relative error among all the contours corresponding to that *hue* interval is compared with the absolute minimum relative error from all the previous *hue* intervals (already processed), saving only the smallest.

- **Detecting the color:** After the sweep is finished, the *hue* interval that has the contour with the least relative error is the one corresponding to the color of the object. The entire ACD algorithm detailing these two steps is presented in Figure 1.

The reason why the contour and the *hue* interval corresponding to the least relative error is considered to be that of the object (object to be detected) lies in the circular shape of the object's projection. In an ideal scenario, the contour of the object's projection will be circular in shape and the minimum inclosing circle for that contour will be the contour itself causing the relative error to be zero.

B. Spherical Object Detection Mechanism (ODM)

Once the color of the object is detected, every subsequent frame is filtered by the *hue* range found by the ACD. The filter consists of a threshold function, which sets a pixel to the value one (white) if it belongs to the *hue* range of the object, and zero (black) in the other case. Further, we compute (after removing noise) the centroid $(\bar{x}, \bar{y})$ of all the white pixels, resorting to the image's spatial moments. The centroid pixel corresponds to the position of the object (center of the spherical object) in the robot's frame, which is the output of the ODM.

$$m_{ji} = \sum_{x,y} I(x,y) x^j y^i$$

$$\bar{x} = \frac{m_{10}}{m_{00}}, \ \bar{y} = \frac{m_{01}}{m_{00}}$$

C. Kalman Filter-based Object Tracking

In order to track the object efficiently and robustly we used a standard Kalman Filter (KF). The ODM is used as the measurement source for the object's position and is used in the KF's update step. For the prediction step, the motion of the robot and that of the object was taken into account. Most of the KF's mathematical details presented here are adapted from [13].

The object's motion is assumed to be a constant velocity plus zero mean Gaussian acceleration noise model and the robot is assumed to move with a constant linear and angular velocities with negligible accelerations. The conversion of the object's velocity from the global frame to the robot frame is done as follows.

Let $v_{\mathrm{o}}^{\mathrm{r}}$ denote the object's velocity in the robot's frame, $v_{\mathrm{o}}^{\mathrm{g}}$ denote the object's velocity in the global frame, v_{r} denote the robot's linear velocity (always in global frame) and ω_{r} denote the angular velocity of the robot about an axis passing through its center of mass and perpendicular to the GP. $p_{\mathrm{o}}^{\mathrm{r}}$ denotes the position of the object in the robot frame. The differentials of these variables are denoted in the dot-format.

$$\begin{cases} v_{\mathrm{o}}^{\mathrm{r}} = v_{\mathrm{o}}^{\mathrm{g}} - v_{\mathrm{r}} - \omega_{\mathrm{r}} \times p_{\mathrm{o}}^{\mathrm{r}} \\ \dot{v}_{\mathrm{o}}^{\mathrm{r}} = \dot{v}_{\mathrm{o}}^{\mathrm{g}} - \dot{v}_{\mathrm{r}} - \dot{\omega}_{\mathrm{r}} \times p_{\mathrm{o}}^{\mathrm{r}} - \omega_{\mathrm{r}} \times v_{\mathrm{o}}^{\mathrm{r}} \end{cases} \tag{1}$$

Since the the robot's and the tracked object's accelerations are assumed to be zero, i.e, $\dot{v}_{\mathrm{o}}^{\mathrm{r}} = \dot{v}_{\mathrm{r}} = \dot{\omega}_{\mathrm{r}} = 0$.

$$\dot{v}_{\mathrm{o}}^{\mathrm{r}} = -\omega_{\mathrm{r}} \times v_{\mathrm{o}}^{\mathrm{r}} \tag{2}$$

To obtain the object's velocity in the robot frame, the robot's velocity adds negatively to the object's velocity in the global frame. An intuitive reasoning is that a static object (implying zero global velocity) would be seen as moving in the opposite direction of the robot's velocity direction when the object is viewed from the robot frame. The robot's angular velocity also affects negatively to the apparent object movement in the robot frame for the same reason.

The state to be estimated by the KF is denoted by $\mathbf{x}$ where $\mathbf{x} = [p_{\mathrm{o}}^{\mathrm{r}} \ v_{\mathrm{o}}^{\mathrm{r}}]^\top = \left[p_{\mathrm{o}x}^{\mathrm{r}} \ p_{\mathrm{o}y}^{\mathrm{r}} \ v_{\mathrm{o}x}^{\mathrm{r}} \ v_{\mathrm{o}y}^{\mathrm{r}}\right]^\top$, which consists of the 2D position and velocity of the object in the robot's frame of reference . The discrete time state transition model and the observation model is given by (3). Henceforth (t) associated to any variable denotes its value at the timestep t.

$$\begin{aligned} \mathbf{x}(t) &= \Phi(t)\mathbf{x}(t-1) + \Gamma(t)u(t) \\ \mathbf{z}(t) &= H(t)\mathbf{x}(t) \end{aligned} \tag{3}$$

where

$$\Phi(t) = \begin{bmatrix} \cos(\Delta\theta) & \sin(\Delta\theta) & \Delta t cos(\Delta\theta) & \Delta t sin(\Delta\theta) \\ -sin(\Delta\theta) & cos(\Delta\theta) & -\Delta t sin(\Delta\theta) & \Delta t cos(\Delta\theta) \\ 0 & 0 & cos(\Delta\theta) & sin(\Delta\theta) \\ 0 & 0 & -sin(\Delta\theta) & cos(\Delta\theta) \end{bmatrix},$$

$$\Gamma(t) = \begin{bmatrix} \frac{-\Delta t sin(\Delta\theta)}{\Delta\theta} & \frac{\Delta t(cos(\Delta\theta)-1)}{\Delta\theta} \\ \frac{\Delta t(cos(\Delta\theta)-1)}{\Delta\theta} & \frac{-\Delta t sin(\Delta\theta)}{\Delta\theta} \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$H(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \tag{4}$$

Δt is the time interval between timestep t and $t-1$, $\Delta\theta$ is the angular displacement and $u(t) = [u_x(t) \ u_y(t)]^\top$ is the linear displacement of the robot between those timesteps acquired

Fig. 2: Perspective lens-based camera mounted on top of a soccer robot

from the robot's odometry measurement. $\mathbf{z}(t)$ denotes the object's observation measurements obtained at the t^{th} timestep from the ODM.

The KF's prediction step and the update step are given by (5) and (6), respectively. The prediction is performed whenever new odometry readings are acquired and the update is performed when the object's measurements are obtained by the ODM.

- *Prediction Step*

$$\begin{aligned} \bar{\mathbf{x}}(t) &= \Phi(t)\mathbf{x}(t-1) + \Gamma(t)u(t) \\ \bar{\mathbf{P}}(t) &= \Phi(t)\mathbf{P}(t-1)\Phi(t)^\top + \mathbf{Q}(t) \end{aligned} \tag{5}$$

- *Update Step*

$$\begin{aligned} \mathbf{K}(t) &= \bar{\mathbf{P}}(t)\mathbf{H}(t)^\top \big(\mathbf{H}(t)\bar{\mathbf{P}}(t)\mathbf{H}(t)^\top + \mathbf{R}(t)\big)^{-1} \\ \mathbf{x}(t) &= \bar{\mathbf{x}}(t) + \mathbf{K}(t)\big(\mathbf{z}(t) - \mathbf{H}(t)\bar{\mathbf{x}}(t)\big) \\ \mathbf{P}(t) &= \big(\mathbf{I} - \mathbf{K}(t)\mathbf{H}(t)\big)\Phi(t)\bar{\mathbf{P}}(t) \end{aligned} \tag{6}$$

The *a priori* and the *a posteriori* error covariance matrices are represented by $\bar{\mathbf{P}}(t)$ and $\mathbf{P}(t)$ respectively. Both were initialized as identity matrices, whereas the process noise covariance matrix, $\mathbf{Q}$, and measurement noise covariance matrix, $\mathbf{R}$, are based on the ODM's measurement and odometry errors. $\mathbf{K}$ denotes the Kalman gain and $\mathbf{I}$ denotes an identity matrix.

IV. Testbed, Implementation and Results

A. *Testbed and Implementation*

Our experimental testbed is the RoboCup Middle Sized League (MSL) where one of the most important prerequisites for the soccer playing robots is to detect and track the soccer ball which is a spherical object. As the official rules of the MSL have evolved, one of the major technical challenges in the recent years is to detect and track an unknown color ball. This makes MSL a suitable choice for implementing and evaluating the method proposed in this article. We used one of our omnidirectional soccer robots for the implementation of the proposed algorithm. The robots acquire new odometry readings every 30 milliseconds. To perform the ball's detection, a perspective lens-based camera (see Figure 2) was fixed on top of the robot at a height of 80cm above the GP and pitched down at an angle of $40°$ w.r.t the GP. Images from this camera were acquired at 30 frames per second (fps). All involved distance computations are based on the perspective camera projection model. An example of the distortion-corrected image from this camera is presented in the Figure 3. All the robot's softwares run on a Sony Vaio laptop, equipped with an Intel Core i3 2.2GHz (quad core) CPU and 4GB of RAM, which is connected to the robot's sensors and actuators through plug-and-play connections (USB and Firewire).

The ACD, ODM and the KF-based tracker were implemented using the robot operating system (ROS) and OpenCV libraries for image processing related processes. Green and white were considered as the dominant color in the background and were filtered out before the ACD was initiated.

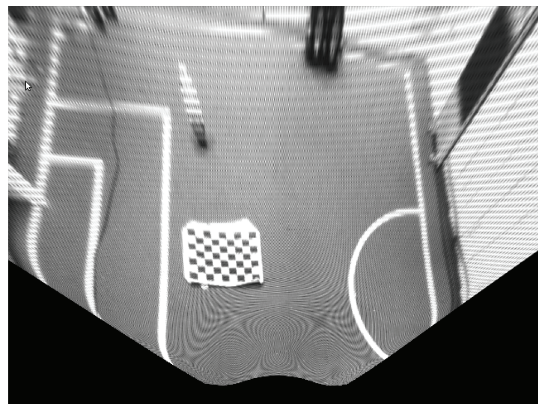

Fig. 3: An example of the distortion-corrected image from the perspective camera.

B. *results*

Experiments were conducted for various balls (note that every ball was uniformly colored, meaning without any significant patterns on it), each of a different color. The robot was able to successfully detect and track the balls upto 6m from itself. Table I shows the range in which the robot was able to detect and track 2 different balls. In Figure 4 we show a series of three images to demonstrate the ACD process. In the first image a red colored ball (color unknown to the detection system) is place in the field of view of the robot's camera. The second image shows result of filtering the background dominant colors: green and white. The third image shows the contours detected on the remaining image, the enclosing circles on those contours and eventually the chosen circle with the minimum relative error corresponding to the actual ball and its color (marked with fluorescent green color on the image).

We further present the statistical results of the KF-based tracker's implementation in case of the red ball for three separate experiments.

Experiment 1: In this experiment the ball was first placed in the field of view of the robot (as it is necessary for the ACD)

Colour	Start Distance (m)	End Distance (m)
Red	2.0	6.2
Yellow	2.0	4.1

TABLE I: Range of tracking achieved by the proposed method on two different balls.

and then moved around the MSL field while the robot was kept static. The robot was able to track the ball successfully even in the case of short-term occlusion. The plots in the Figure 5 present the ball's estimated distance to the robot during the experiment and the KF's innovation over time. The region marked *Occlusion* in the plot of Figure 5 shows the sharp increase in the filter's innovation during the period in which the ball is occluded from the camera's field of view.

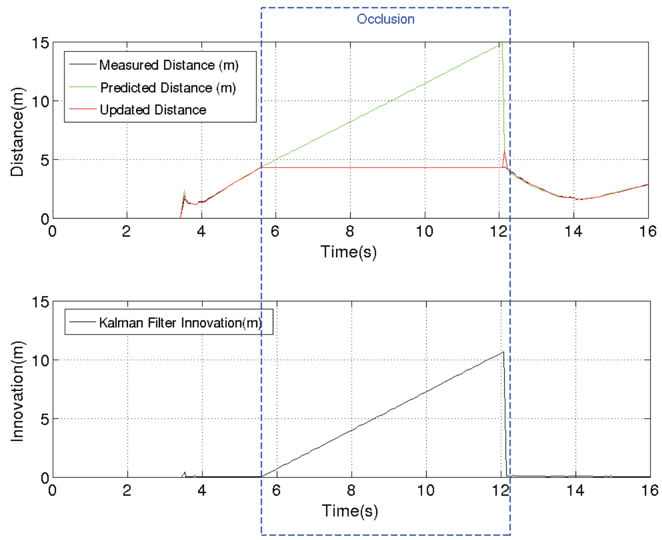

Fig. 5: Results of **Experiment 1** where only the ball was moved while the robot was static.

Experiment 2: In this experiment the ball was kept static while the robot was rotated around its central axis perpendicular to the GP so that initially the ball is in the robot's camera's field of view, loses it during the robot's rotation and eventually gets it back in the field of view. Results of this experiment is plotted in the Figure 6 where we show the estimated distance to the ball from the tracking robot, the bearing to the ball w.r.t the robot's heading and its corresponding residual referred to as the KF's angular innovation.

Experiment 3: In this experiment both the ball and the robot were moved, however, the ball never left the camera's field of view. Results of this experiment are plotted in the Figure 7 which show both the estimated distance to the ball from the robot and the filter's innovation. From this results we infer that the innovation was slightly noisy (variance of ~ 7 cm^2) but with a low mean of ~ 3 cm. The reason behind the higher variance lies in the constant velocity plus zero mean Gaussian acceleration noise model used in the KF for the ball's motion update. To cope with the erratic movements of the ball, that causes the spikes in the corresponding innovation plot, a more sophisticated motion model would be required, e.g, a mechanism that switches between different motion models depending on the predicted trajectory of the tracked object.

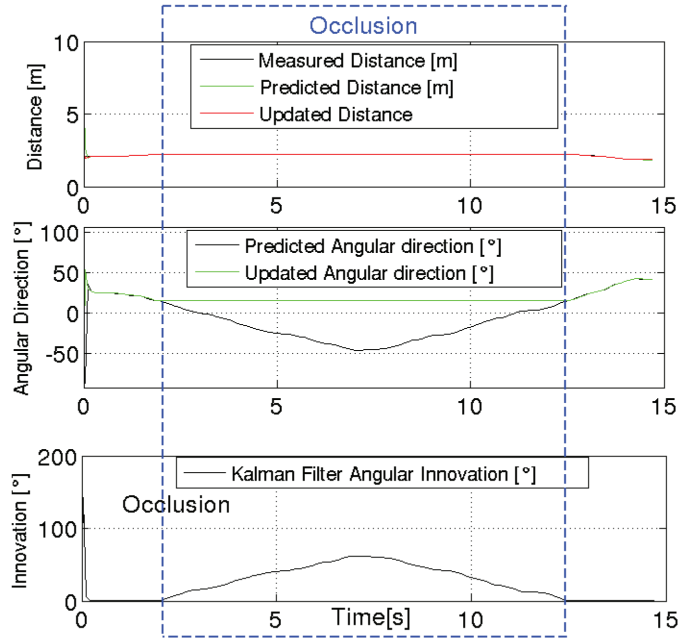

Fig. 6: Results of **Experiment 2** where the ball was static while the robot was rotated around its central axis perpendicular to the ground plane.

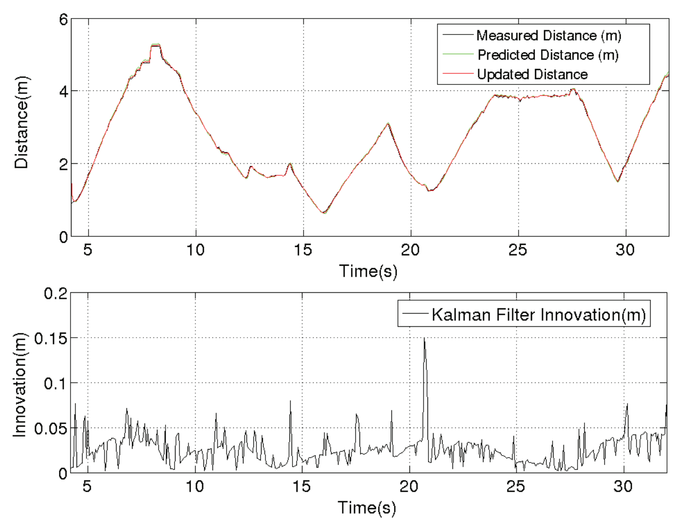

Fig. 7: Results of **Experiment 3** where both the ball and the robot move.

a) Unknown-color ball in the robot's view b) Background dominant colors filtered c) Ball color detected (marked as fluorescent green)

Fig. 4: Demonstration of the automatic color detection (ACD) process.

V. Conclusions and Future Work

In this article we presented a novel method to automatically detect the color of a spherical object before detecting and tracking it using a Kalman filter-based tracker. The fast execution speed of the color detection method enables it to be executed periodically while running the tracker to cope with the changes in the lighting conditions of the environment. The method was implemented on the soccer playing robots to track unknown-color soccer balls with successful results. A few points that could be enumerated for the purpose of future work are as follows.

- In order to deal better with occlusions and to have a smoother trajectory of the tracked object, cooperation among multiple robots and innovative motion models are required, e.g, an alpha-beta filter [14]. It is a steady-state form of the nearly constant velocity filter. Since an erratically moving object is affected by random positive-mean acceleration, a good object motion model needs to take this effect into account.
- Although the camera is pitched down in our application, it still detects some irrelevant areas of the environment. A possible future improvement would be to choose a configuration where the camera's field of view is better optimized to make use of the maximum possible image space.

Acknowledgment

This work was funded in part by *Fundação para a Ciência e a Tecnologia* (ISR/IST pluriannual funding) through the PIDDAC Program funds.

References

[1] C. Lemaitre, M. Perdoch, A. Rahmoune, J. Matas, and J. Miteran, "Detection and matching of curvilinear structures," *Pattern Recognition*, vol. 44, no. 7, pp. 1514 – 1527, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320311000148

[2] O. Carmichael and M. Hebert, "Shape-based recognition of wiry objects," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 2, june 2003, pp. II – 401–8 vol.2.

[3] V. Ferrari, T. Tuytelaars, and L. V. Gool, "Object detection by contour segment networks," in *Proceeding of the European Conference on Computer Vision*, ser. LNCS, vol. 3953. Elsevier, June 2006, pp. 14–28.

[4] M. Donoser, S. Kluckner, and H. Bischof, "Object tracking by structure tensor analysis," in *Pattern Recognition (ICPR), 2010 20th International Conference on*, aug. 2010, pp. 2600 –2603.

[5] M. Taiana, J. Santos, J. Gaspar, J. Nascimento, A. Bernardino, and P. U. Lima, "Tracking objects with generic calibrated sensors: An algorithm based on color and 3d shape features," *Robot. Auton. Syst.*, vol. 58, no. 6, pp. 784–795, Jun. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2010.02.010

[6] M. Wenig, K. Pang, and P. On, "Arbitrarily colored ball detection using the structure tensor technique," *Mechatronics*, vol. 21, no. 2, pp. 367 – 372, 2011, ¡ce:title¿Special Issue on Advances in intelligent robot design for the Robocup Middle Size League¡/ce:title¿. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957415810001297

[7] D. Gohring and H.-D. Burkhard, "Multi robot object tracking and self localization using visual percept relations," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, oct. 2006, pp. 31 –36.

[8] A. Ahmad and P. U. Lima, "Multi-robot cooperative object tracking based on particle filters," in *Proc. of the European Conference on Mobile Robots (ECMR 2011)*, Örebro, Sweden, Sep 2011.

[9] J. Messias, M. Spaan, and P. Lima, "Multi-robot planning under uncertainty with communication: a case study," in *Multi-agent Sequential Decision Making in Uncertain Domains*, 2010, workshop at AAMAS10.

[10] J. Matas, C. Galambos, and J. Kittler, "Progressive probabilistic hough transform," 1998.

[11] D. A. M. Antnio J. R. Neves and A. J. Pinho, "A hybrid vision system for soccer robots using radial search lines," in *Proc. of the 8th Conference on Autonomous Robot Systems and Competitions, Portuguese Robotics Open - ROBTICA'2008*, Aveiro, Portugal, 2008.

[12] H. Lu, H. Zhang, J. Xiao, F. Liu, and Z. Zheng, "Arbitrary ball recognition based on omni-directional vision for soccer robots," in *RoboCup 2008: Robot Soccer World Cup XII*, ser. Lecture Notes in Computer Science, L. Iocchi, H. Matsubara, A. Weitzenfeld, and C. Zhou, Eds. Springer Berlin Heidelberg, 2009, vol. 5399, pp. 133–144. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02921-9_12

[13] Z. Marinho, J. Messias, and P. U. Lima, "Multi-object tracking based on histogram classifier and kalman filtering," in *ISR/IST Internal Report*, 2012.

[14] W. D. Blair, "Design of nearly constant velocity track filters for tracking maneuvering targets."

978-989-97531-2-9