Real-Time Tracking of Mobile Robots in Structured Environments

Rafael García, Joan Batlle, and Marc Carreras

Computer Vision and Robotics Group
Institute of Informatics and Applications
Dep. of Electronics, Informatics and Automation
University of Girona, Av. Lluis Santaló, s/n, E-17003 Girona (Spain)
Tel. +34.72.418474 Fax +34.72.418098
e-mail: {rafa,jbatlle}@eia.udg.es

Abstract

In this paper a vision system is used to control a set of robots in an industrial environment. The algorithm for tracking the moving robots using colour features is described. An extended Kalman filter is applied to filter out some of the noise and to estimate recursively the position of the robots. As we want to obtain nearly video-rate performance, we have developed a specific hardware for image segmentation and object-tracking.

Keywords: Real-time Tracking, Kalman Filter, Colour, Structured environments

1. INTRODUCTION

Tracking moving robots over an industrial environment is a complex problem in computer vision when one wants to obtain nearly video-rate performance. Specifically, the problem to be resolved is controlling several moving robots moving in a structured environment. Perception and computation are performed off-board the robot, thus a global view of the "world" is available, simplifying the pose detection of the objects. Figure 1 shows a scheme of path planning in an industrial environment.

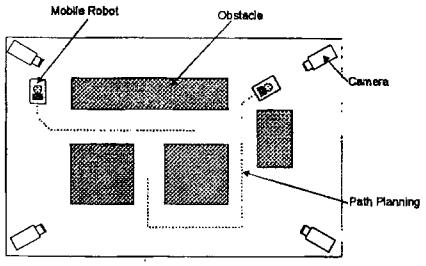

Figure 1. Supervision of mobile robots in an industrial environment.

It has been proved that fast spatial sensing is very important for this kind of environments. As we want to find the location of every robot on the scenario, the correspondence problem has to be addressed. We have to establish temporal reciprocity: given a robot A in image i_k, it may be put in correspondence with any robot B in the image i_{k+1}. To solve this difficulty, we use 2 constraints in order to reduce the amount of potential matches for any given robot from the image i_k. First, minimum distance criteria can be applied [1]: the detected position of an object in image i_k is matched with the closest position detected in image i_{k+1}. Secondly, if every robot is

fitted with a different-coloured top, the chromatic characteristics can help to solve the correspondence [2]. We propose a reliable system which can solve this correspondence by means of an accurate use of the colour attributes.

The paper is organised as follows: in the first part of the paper, we describe the developed platform for testing our system in the lab. Then, section 3 explains how the vision system performs segmentation and tracking of the robots over time. An algorithm for real-time location of the objects is explained. Next, the results are shown through some test images of the system. Finally, the some conclusions are exposed and the future work is detailed.

2. THE LAB TESTING ENVIRONMENT

Since our aim is to develop a system for tracking mobile robots in a structured environment, a lab-testing platform has been developed. In this platform, a global vision system is used to track the robots on a field. This system must track several robots, detecting their position and orientation. First, the output of the overhead camera is fed into one of the off-board computers and the vision input is processed. Once this image has been examined, the coordinates of the positions of the robots are passed to the control system. It uses the vision system as a sensor to schedule a strategy, communicating to the robots by means of a two-way radio-link. Thus, strategical navigational control commands are sent to the robots. By using this wireless link, the central control system has access to local sensing placed on each robot, i.e. IR proximity sensors, motor encoders, etc. Then, the accuracy of the control feedback loops is increased by additional information sent by the robots.

In order to locate the robots, special colour markings are placed on the top of the robots. A single colour patch could be enough to provide orientation information as was shown in [3]. However, it implies a high computational cost and the need of a specific hardware to achieve real-time performance. Therefore, in order to detect orientation, we use a two-colour patch for every robot. One of the colours indicates the robot's position, while the second one gives a measure of its orientation, in this way, we are able to know if the robot is facing forward or backwards.

The motivation for using colour tracking is, first of all, because of its simplicity and low computational cost; and, secondly, because regions provide more robust motion estimations than contours [4,7]. Therefore, the proposed system uses a region-based tracking algorithm which should achieve real-time performance.

3 TRACKING ROBOTS OVER TIME

The tracking module can be divided into three sub-tasks: region labelling, blobs detection and motion prediction.

3.1 Region Labelling

The aim of this module is to isolate the mobile objects from the rest of the image, assigning a different label to every region to be tracked.

As we want to classify the objects by using their chromatic features, a real time colour conversion has to be performed. We have chosen the models showed in equations (1) and (2), because of their scale-invariance properties.

$$H(r,g,b) = tan^{-1}\frac{\sqrt{3}(g-b)}{(r-g)+(r-b)} \quad (1)$$

$$S(r,g,b) = 1-\frac{3\ min(r,g,b)}{r+g+b} \quad (2)$$

From the previous equations it can be seen that the conversion from *RGB* to *HS* has the property of scale-invariance, that is, $H(r,g,b) = H(\alpha r, \alpha g, \alpha b)$ and $S(r,g,b) = S(\alpha r, \alpha g, \alpha b)$. Thereon, hue and saturation are stable under variations on the intensity of the illuminant [6]. However, this linear behaviour does not hold when one of the RGB components reaches its maximum value, known as colour clipping in the literature.

As computing H and S for every pixel in an image is computationally expensive, we first load a Look Up Table (LUT) into the host RAM, storing 2^{18} values of hue, according to equation (1). Then, the hue of all the combinations of *RGB* using 6 bits per channel are available just in a read-cycle of the host. A second LUT is loaded with all the combinations of saturation also using 6 bits per channel. In order to achieve the right threshold for each colour component, an overhead image of the field is acquired. Next, we mark interactively on this image a region belonging to every one of the objects to be tracked. Then, the arithmetical mean and the standard deviation of this zones are calculated for hue and saturation, in order to choose the right thresholds. By using these parameters ($[H_{min},H_{max}]$ and $[S_{min},S_{max}]$ for every region), a "labelling table" is computed. This table consists of 2^{18} positions of 8 bits, that is, an 8-bit word for every *RGB* combination. We will call this 8-bit word the "*object label*", and its value will codify the chromatic components of every object in the scene. If the corresponding *RGB* combination belongs to the hue and saturation intervals of object A, then the object label will contain the code "*1*". In the case of object B, this code will be "*2*", and so on for all the objects in the scene. When the *RGB* combination does not belong to any of the objects to be tracked, then the segmentation word will store "*0*". This way the objects are labelled depending on their colour attributes. It should be noted that the computation of the labelling table is performed off-line, in the set-up process. Once the colours have been characterised the system will segment and label using this algorithm in real time.

3.2 Pose detection

As each of the team robots is equipped with a different colour top, we can control the robots associating a distinct label to every robot. In the set-up process, an initial estimate of the location of the robots and the ball is computed. Then, we use a minimum distance approach, using this estimate as the most probable location of this object in the next image. Several, works in robotics have used before this idea [1,5]. Then, the detection module just searches for every object in a small window, which is adapted to the kinematics of the object to be tracked. In the next image, the search window is centred at a different position, and the centre is estimated by the prediction module, as explained in section 3.3.

3.3 Prediction

Robots are guided systems, without an a-priori well defined dynamics. However, they frequently behave as a dynamic system, allowing the development of a motion model. Thence, an Extended Kalman Filter (EKF) has been developed in order to filter and predict the position of the robots in the future time instants. Normally, the detection of the robots' location is noisy, and its motion is of non-linear nature. Thus, a non-linear predictor is adequate for our system [8]. The equations of the Extended Kalman Filter are described below, but a more detailed description of Kalman-Bucy Filters can be found in [9].

The Extended Kalman Filter is a recursive estimator for estimating the state of a non-linear system. In our implementation the state will be denoted by a 4-dimentional vector $\hat{x}_k^T = [x, y, \dot{x}, \dot{y}]$, where x and y denote the position of the ball in the image, and $\dot{x}$ and $\dot{y}$ are the velocities in the respective directions. The measurements from the latest image are stored in z_k. Just x and y positions are obtained from the image. Gain matrix K_k relates the amount of influence the error between the measurement z_k and the previous estimation x_k^-. So, it measures how match the filter relies on the current observations z_k, through the update equations:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1} \quad (3)$$

$$\hat{x}_k = x_k^- + K_k (z_k - h(x_k^-, 0))^{-1} \quad (4)$$

It can be seen from equation (3) that the gain matrix K_k is computed by using the predicted error of those elements in the state parameter vector that are obtained from the image. H_k gives the noiseless connections between the measurement and the state vector at time t_k, and R_k is the measurement error. Then, equation (4) updates the estimate with the measurement. The difference between the estimated and measured parameters can be considered to be a prediction error, which is caused by either faulty measurement, faulty prediction, or a combination of both. A proportion of the predicted error is added to the parameter estimate $\hat{x}_k^-$ to produce an updated state parameter vector $\hat{x}_k$, depending on the values of K_k.

Next, updated error covariance is computed:

$$P_k = (I - K_k H_k) P_k^- \tag{5}$$

where the portion of the gain matrix that is associated with elements obtained from the image (ball position) is substracted from an identity matrix to yield a proportion of non-gain. This is multiplied with the estimated error covariance to produce an updated error covariance, reflecting the remaining uncertainty about the state parameters. This implies that as K_k decreases, the estimated error used to update P_k increases proportionally.

Equations (3), (4) and (5) perform the update process of the filter. They use the current measurement to refine the current estimate and recompute the error covariance. Equations (6) and (7) perform the propagate process, projecting ahead for parameter and error covariance at the next time step.

$$\hat{x}_k^- = f(\hat{x}_k, u_k, 0) \tag{6}$$

$$P_{k+1}^- = A_k P_k A_k^T + W_k Q_k W_k^T \tag{7}$$

These propagate equations are normally applied n times in order to obtain a prediction of the system's state in n time-steps ahead. Normally, the detection of the robots' location is noisy, and its motion is of non-linear nature. Thus, a non-linear predictor is adequate for our system [4]. However, in a small interval of time we can consider that the motion is linear, using a constant velocity model, as shown in equations (8–11):

$$x_{k+1} = x_k + \dot{x}_k \cdot \Delta t \tag{8}$$

$$y_{k+1} = y_k + \dot{y}_k \cdot \Delta t \tag{9}$$

$$\dot{x}_{k+1} = \dot{x}_k \tag{10}$$

$$\dot{y}_{k+1} = \dot{y}_k \tag{11}$$

Tests have shown that the prediction line remains stable at the propagation step. Then, we propose the use of the propagate equations a short number of times, in order to obtain a regression line which can be applied ahead to find the prediction in n time-steps ahead.

The lab-testing robots are designed to get up to 0.6 m/s robotic movements and to control them, in the sense of dynamic systems, requiring advanced control technology. Work has been done on specialising the approach of automatic control community by introducing the classical structure behaviour-supervision-control algorithms [10,11]. Planning of coordinated motions for cooperating robots opens a new deal in motion planning. This problem is quite challenging due to the fact that the situation of the robots is continuously varying, that is, this kind of application consists of a dynamic system which constantly evolve.

The robots are devised and controlled taking into account two main tasks: first, low-level tasks, which aim is to execute the movement commands with accuracy. This level consists of a velocity closed-loop control implemented on the robots. Secondly, high-level tasks are performed, consisting in generating the movement commands to be executed by the robots, where data provided by the vision system are used.

3.4 Hardware Implementation.

In order to achieve real time performance, the region labelling and pose detection modules have been implemented using Hardware Description Languages (VHDL), and synthesised on an Altera Flex-10K100 Field Programmable Gate Array (FPGA). The developed hardware is shown in figure 2.

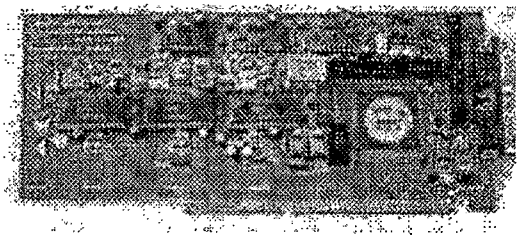

Figure 2. Real-time tracking processor.

The tracking system incorporates a connector for reprogramming the FPGA in-system, facilitating the fast prototyping and development of algorithms in hardware. Thus, future improvements of the system can easily be carried out. Moreover, digital cameras can be plugged into the processor through the digital video connectors, allowing to process images at a higher frame rate.

4. EXPERIMENTS AND RESULTS

The system is granted with a few test programs, which supply information for checking the vision system. These tests can be used to debug the system, usually searching for bad initialisation or calibration parameters. Some tests are shown in figure I. First, we can see the result of the segmentation and labelling process. Sometimes, and due to a bad illumination, the threshold levels (automatically computed in the set-up process) for H and S are too narrow to provide a good segmentation over the whole field. Secondly, we can check whether the search area, centred at the next predicted position, is big enough to keep the tracked object inside its window. Next, the prediction of future locations of the ball by means of the EKF can be tested with the last image illustrated in figure I.

Finally, the tracking results are drawn in a window, for obtaining a visual feedback of the system performance. With this approach, a constant frequency of 50 processed images per second is achieved.

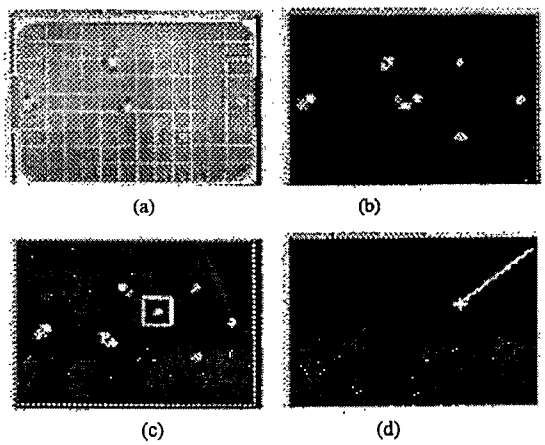

Figure 3. Test images: (a) Original image; (b) Segmented image assigning a different label to every region to track. (c) Position of the search window using a linear first order prediction. (d) Prediction of the robot position 10 time instants forward, using the EKF

5. CONCLUSIONS

We have presented a system to track several objects at video rate. Significant improvements have been introduced since our team participated in last year's robotic competitions. The system tracks up to 5 moving robots at video rate (50Hz in the PAL standard) thanks to the throughput of our specific hardware, Providing the coordinates of the centroids corresponding to every robot independently of their motion. The use of hue and saturation has been proved to be a robust framework under illumination changes. An accurate prediction can be achieved through an extended Kalman filter.

6. REFERENCES.

[1] Han, K., and Veloso, M., "Reactive visual control of multiple non-holonomic robotic agents," in Proceedings of IEEE International Conference on Robotics and Automation, 1998.

[2] Oller, A., García, R., Ramon, J.A., Figueras, A., and de la Rosa J. Ll. "Cooperation Among Robots by Means of Multi-Agent Decision Making Framework under MATLAB/ SIMULINK" in Proceedings on Micro-Robot World Cup Soccer Tournament, pp. 37–45, Taejon, Korea, 1997.

[3] Sargent, R., Bailey, B., Witty, C., and Wright, A. "Fast vision tracking and coordinated control for soccer-playing robots" in Proceedings on Micro-Robot World Cup Soccer Tournament, pp. 59–65, Taejon, Korea, 1997.

[4] Bouthemy, P., François, E., "Motion segmentation and qualitative dynamic scene analysis from an image sequence," International Journal of Computer Vision, pp. 157–182, vol. 10:2, 1993.

[5] Kim, J-H., Shim, H.-S., et al. "Cooperative multi-agent robotic systems: from the robot-soccer perspective", in Proceedings on Micro-Robot World Cup Soccer Tournament, pp.3–14, Taejon, Korea, 1997.

[6] Perez, F., and Kock, C., "Toward color image segmentation in analog VLSI: Algorithm and hardware," International Journal of Computer Vision, vol. 12, pp.17–42, 1994.

[7] Bascle, B., Deriche, R., "Région tracking through image sequences," INRIA Research Report no. 2439, 1994.

[8] Han, K., and Veloso, M., "Physical model based multi-objects tracking and prediction in RoboSoccer," in Working Notes on the AAAI Fall Symposium on Model-directed Autonomous Systems, Cambridge, MA, 1997.

[9] Kalman, R. E., "A new approach to linear filtering and prediction problems," Trans. ASME, Journal of Basic Engineering, pp. 34-45, 1960.

[10] de la Rosa, J. Ll., Oller, A., Vehi, J., and Puyol, J., "Soccer team based on agent-oriented programming" Robotics and Autonomous Systems, vol 21, pp. 167–176, 1997.

[11] Oller, A., de la Rosa J. Ll., García, R., Ramon, J.A., and Figueras, A., "Micro-robots playing soccer games: a real implementation based on a multi-agent decision-making structure" Intelligent Automation and Soft Computing, in press, 1999.