In: Mobile Robots: New Research
Editor: John X. Liu, pp. 119-143

ISBN: 1-59454-359-3
© 2005 Nova Science Publishers, Inc.

Chapter 3

VISION SYSTEM DESIGN FOR MOBILE ROBOT TRACKING

Gregor Klančar*[*], *Marko Lepetič*, *Matej Kristan*, *Rihard Karba
University of Ljubljana, Faculty of Electrical Engineering, Tržaška 25,
SI-1000 Ljubljana, Slovenia, Tel: +386-1 476 8701, Fax: +386-1 426 4631

Abstract

This chapter introduces a complete thematic of vision system for mobile robot tracking and control. It consists of a global vision system for estimation of positions and orientations of mobile robots and methods for improvement of bad operating conditions such as noise, camera lens distortion and non-uniform illumination.
The basic operation of a vision system is divided into two steps. In the first, the incoming image is scanned and pixels are classified into a finite number of classes. At the same time, a segmentation algorithm is used to find the corresponding regions belonging to one of the classes. In the second step, all the regions are examined. A selection of the ones that are a part of the observed object is made by means of simple logic procedures. The novelty of the used approach is focused on optimization of the processing time needed to finish the estimation of possible object positions.
Further on an approach to improve an already existing vision system performance under bad operating conditions is presented. Some fundamentals and solutions to accompanying problems in vision system design for mobile robot tracking are presented. Besides methods for filtering and improvement of identified noisy data the two main factors which deteriorate the performance are dealt with, namely, non-uniform illumination and camera lens distortion. For the former the problem area and its origins are focused on and a solution for its compensation by applying multiplicative component defined by illumination plain is given. The latter consists of two steps. In the first, radial lens distortion fundamentals are discussed. The suggested solution for its verification is realized by a geometry model of lens projection. The second step covers the perspective distortion originating from the tilt of the camera. For its correction an efficient and robust method of vanishing point detection is applied. Both correction methods contribute to a vision system performance if implemented in the appropriate manner.

[*] E-mail address: gregor.klancar@fe.uni-lj.si

Applicability of the presented approaches is confirmed on a robot soccer test bed. Robot soccer is a fast dynamic game and therefore needs an efficient and robust vision system. To improve the results of the robot soccer vision system the proposed camera calibration and non-uniform illumination correction algorithm are implemented. The lens correction method successfully corrects the distortion caused by the camera lens, thus achieving a more accurate and precise estimation of the object position. The illumination compensation improves robustness to irregular and non-uniform illumination which is nearly always present in real conditions.

1 Introduction

There are many ways of detecting moving objects using color cameras. However, the vision systems based on color information proved to be more simple, robust and faster than most of other recognition methods as stated in [3,5,13,16]. Sargent *et al.* [13] developed a fast real-time vision system with the aid of a special hardware accelerated system, which only makes sense when software optimizations or accelerations are not possible. A more reliable vision tracking of moving objects can be achieved by using robust statistics and probability distributions. A good example of the latter is given in the color-based face tracking implemented by Bradski [2]. Bruce *et al.* [3] suggested a fast vision system for mobile robots by means of efficient color segmentation and a two-pass connected region determination algorithm. Another important contribution to the robot soccer vision design was introduced by Wyeth *et al.* [16], with special consideration given to the robustness of varying playground illumination conditions. Most of the approaches try to classify the pixels of an image into one of a predefined number of classes. The most common are: linear color thresholding, *K*-nearest neighbor classification, neural net-based classifiers, classification trees and probabilistic methods [10,1,8].

The chapter presents a design of a global vision system for estimating current object positions and orientations on the playground. The MiroSot category soccer robots we are interested in are without on-board position sensors. Thus a precise and fast global vision has to be designed for robots control and navigation in a partially controlled, dynamically changing environment. When designing the vision system, the following requirements have to be accomplished:

- computational efficiency,
- high reliability,
- good precision, and
- robustness to noise, non-uniform illumination and different color schemes.

The last characteristic is essential for the system to function well when using it under different conditions present at competitions [16].

In this paper, a fast approach with constant thresholding and back-stepping algorithm is presented, where a special attention is given to the efficiency aspect. The thresholds can be presented as boxes in 3-dimensional color spaces. These thresholds are determined by means of off-line learning. If an incoming pixel color falls inside one of the predefined boxes, then it is classified as belonging to the class associated with this box. This first step is followed by the second step where the pixels belonging to one class (a connected region) are distinctively labeled. With the main purpose of obtaining all fully connected regions, a back-stepping algorithm is applied. Both steps are done with just one scan of the image. Then the logic part and a simple optimization method are employed to select the proper regions from the previously generated ones. After this logic the positions and orientations of the objects on the playground are estimated. To improve results of the vision system the camera calibration and non-uniform illumination correction algorithm are implemented. The former corrects distortion caused by the camera lens, thus achieving a more accurate and precise objects positions estimation, while the latter improves robustness to irregular illumination and non-uniform illumination conditions.

An attempt to improve the already-existing vision systems performance under poor operating conditions is next presented. Two main factors, which adversely affect performance, are dealt with: non-uniform illumination and camera lens distortion. For the former, the focus is placed on the problem area and its origins [6], with a solution for its compensation by means of the application of a multiplicative component defined by an illumination plane given. The latter consists of two steps. In the first, radial lens distortion fundamentals are discussed [9,14]. The suggested solution for its verification is realized by means of a geometric model of lens projection [11]. The second step covers the perspective distortion originating from the tilt of the camera. For its correction, an efficient and robust method of vanishing point detection [4,12] is applied. The applicability of the presented approaches is confirmed on the robot soccer test bed. To improve the results of the robot soccer vision system [5], both the proposed camera calibration and non-uniform illumination correction algorithm are implemented.

The chapter is organized as follows. In section 2 a brief overview of the system is given. The method used for pixel classification is explained in section 3. Section 4 focuses on the algorithms for image segmentation and region labeling. The algorithm for object estimation is illustrated in section 5. Section 6 resumes the data filtering, camera calibration and non-uniform illumination correction implementation. Obtained experimental results are shown in section 7. The chapter ends with conclusions and some ideas for future work.

2 System Overview

The presented vision system is demonstrated on a robot soccer set-up. The soccer robot set-up, Fig. 1, consists of ten MiroSot category robots (forming two teams) of size 7.5 cm cubed, a rectangular playground of size 2.2×1.8 m, a digital color camera Sony DFW-V500, and a personal computer Pentium 4. The vision part of the program processes the incoming images, of a resolution of 640×480 pixels, to identify the positions and orientations of the robots and the position of the ball. Each robot has two square-shaped color patches (Fig. 2). One is the team color and the other is the identification color patch. According to FIRA (Federation of International Robot-soccer Association) rules, the team color must be blue or yellow, the ball must be orange and identification colors can be any color except the team and ball color. The vision algorithm finds objects on the playground by taking their color and shape into consideration. If an incoming pixel color falls inside one of the predefined boxes (defined by thresholds), it is classified as belonging to the class associated with this box. The thresholds are presented as boxes in three-dimensional color spaces. The pixels belonging to one class (a connected region) are then distinctively labeled. The logic part and a simple optimization method are employed to select the proper regions from the previously generated ones. Finally, the control part of the program calculates the linear and angular speeds, that the robots should have in the next sample time according to the current situation on the playground. These reference speeds are sent to the robots by a wireless connection and they start moving according to the received commands. The above-mentioned cycle repeats itself 30 times per second.

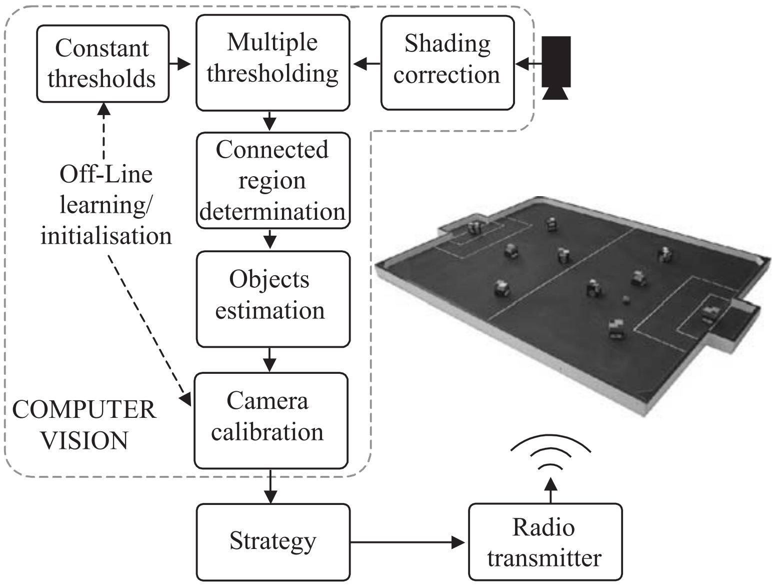


Fig. 1. System overview.

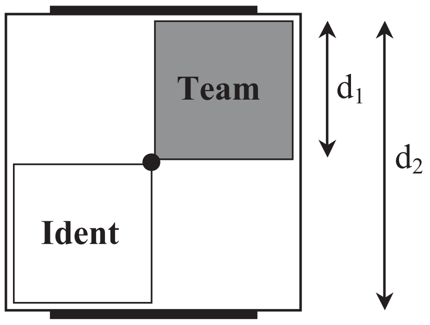


Fig. 2. Color patches on the robot.

3 Pixel Classification

To enable detection of different color patches, each pixel has to be classified into one of the predefined colors.

3.1 Color Spaces

The color image can be presented by the use of different color spaces such as RGB, HSI, YUV and others. By using the RGB space, the regions with the same color are best presented in a three-dimensional color space with conical volumes. When using simple thresholds for pixel classification, HSI and YUV color spaces are most appropriate. They code the information about chrominance in two dimensions (H and S or U and V) and only one dimension includes the information about intensity (I or Y). With these color spaces, any particular color on the playground can be described with wide areas between thresholds in the intensity dimension, while the threshold areas for other chrominance dimensions are narrow. Also these color spaces are more robust to different illumination conditions.

Both RGB and YUV spaces were used in our experiments with constant thresholds. The classification results obtained were similar. However, to obtain the YUV color representation, a transformation from the original RGB space had to be done. This transformation was time consuming although the optimization by using look-up tables was used. Including the optimization it requires some 30 ms, while the rest of the program takes only 8 ms to identify objects from the image. Therefore, YUV or any other color space should be used only when it can be directly obtained from the frame grabber. In the application the digital camera outputting RGB color space was used with image scan time of 33 ms (30 Hz).

3.2 Thresholding

The basic idea is to classify each pixel according to the preset color thresholds of each object. This can be done using the following code

```
for i=1 to number of colors on playground
if (R >= R_lower_boud AND R <=R_upper_bound AND
    G >= G_lower_boud AND G <=G_upper_bound AND
    B >= B_lower_boud AND B <=B_upper_bound )
Pixel_color =i-th color;
else Pixel_color=background;
end
```

This simple part of the code requires 6 relational and 5 AND operations for each pixel classification. This code is repeated for each pixel and color we want to classify.

To improve this operation, *i e*. to check all colors at the same time, the idea of parallelism by means of a look-up table is considered. Three N×32-bit integer arrays are allocated, where *N* corresponds to the number of color levels (usually *N*=256) and the maximum number of colors to be classified is 32 respectively (Fig. 3). Each bit in a 32-bit memory location is associated with one color. Although the algorithm is able to classify 32 different colors, only 13 are enough for the purpose of the robot soccer game. Because the computer has a 32-bit arithmetic, the computational burden is the same as for the 16-bit memory location only.

Let us suppose that we want to classify a yellow patch with the color values in the following range

```
200 <= R <=220
230 <= G <=250
10 <= B <= 30
```

Suppose this color is associated with the 31st bit of each memory location. So in this case the highest bits in the memory locations between 200 and 220 in *RClass*, 230 and 250 in *GClass* and 10 and 30 in *BClass* are set to 1 respectively. The same procedure is then performed for bits 0-30 for other color patches.

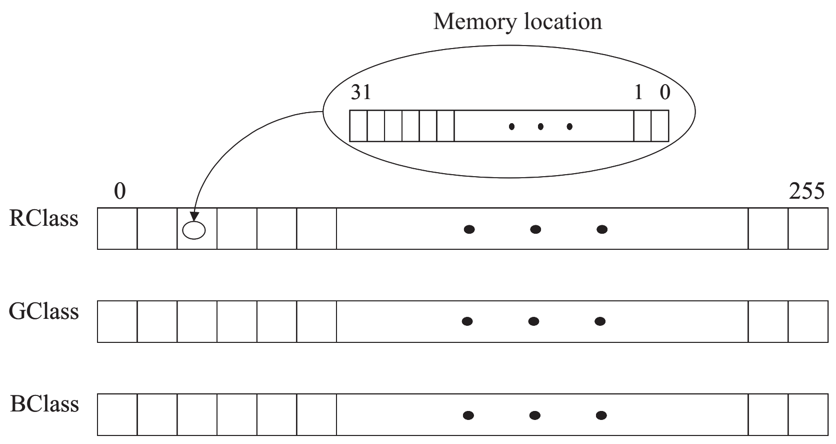


Fig. 3. Look-up tables for pixel classification.

At the run time, memory locations with the index corresponding to the current pixel *R*, *G* and *B* values are taken. The bitwise AND operation between the chosen memory locations gives the information about the classification of the pixels. If the result has the 31st bit set to 1, then the pixel is classified as yellow.

With this methodology, a multiple thresholding (the thresholds for all color patches checked at the same time) is made in only one scan of the image. As the multiple threshold operation takes just two AND operations, it significantly reduces computational burden.

4 Image Segmentation and Component Labeling

To estimate patch positions, first all identification patches and the regions belonging to the ball, team and opponent team patches have to be located. The number of those regions on the playground (*K*) can be higher than the number of all patches due to camera noise. Image segmentation in *K* regions and labeling is done fulfilling the following five conditions [10]:

- $\bigcup_{i=1}^{K} R_i = R$,
- $R_i \cap R_j = \varnothing$, $i, j = 1, 2, \ldots, K$, and $i \neq j$,
- R_i is a connected region of pixels,
- $P(R_i) = 1$, $\forall i$,
- $P(R_i \cup R_j) = 0$, $i \neq j$, and R_i, R_j are neighbors,

where $P(x)$ is a logical predicate, which takes value 1 if all the pixels of the region accomplish a criterion of homogeneity. In our case, the homogeneity criterion is the equality in color.

According to the first two conditions, the regions R_i together must occupy the entire image R and the regions must not have common pixels. Due to the third condition, there must be at least one path of pixels of the same color connecting any two pixels in the region. Moreover, the regions must be homogeneous with respect to the color. Additionally, the neighboring regions must not be of the same color, as stated in conditions four and five.

4.1 The Algorithm

The sequential connected component labeling algorithm, originally developed by Rosenfield and Pfaltz, [17] is a well known technique for efficient image segmentation. It requires two passes through the image. In the first all pixel labels are generated with equivalent labels being stored. The second pass replaces each label with its representative label. A number of researchers tried to improve the efficiency of the above algorithm mostly by optimizing the second pass of the algorithm. Our approach is presented in the sequel, focusing on one pass variant only. Pixel classification and image segmentation are merged by the aid of a corresponding algorithm. Its main property is that the image classification and segmentation is done with only one pass (row-vice) through the image, which considerably contributes to time efficiency. The results of the mentioned algorithm are labeled regions with the following information:

- region number,
- color,
- number of pixels belonging to this region,
- pointers to each pixel belonging to this region,
- coordinates of the center of the region, x_{avg} and y_{avg}.

Prior to the start of the algorithm, the color thresholds are selected for each component to be identified: ball, robot 1, …, robot 5, team, opponent robots and opponent team.

Connected component labeling algorithm:

- The algorithm starts analyzing the 1st pixel of a given region of interest (in our case the whole image).
- If the color of the pixel under study is a valid color and at the same time different from the color of the upper and left neighbor pixels, a new region is created [Fig. 4(a)].
- If the color of the pixel under study is a valid color and it is also equal to the color of the upper or left pixel, then the pixel under study is added to the region of the upper or left pixel [Fig. 4 (b)].
- If the color of the pixel under study is a valid color and at the same time equal to the upper and left pixel colors, then [Fig. 4 (c)]:

    - if the upper and left pixels belong to the same region, the pixel under study is added to this region,

- otherwise, the pixel under study is added to the region with a bigger number of pixels, the pixels of the region with lower quantity of elements are copied to the bigger region, and then the region is deleted.

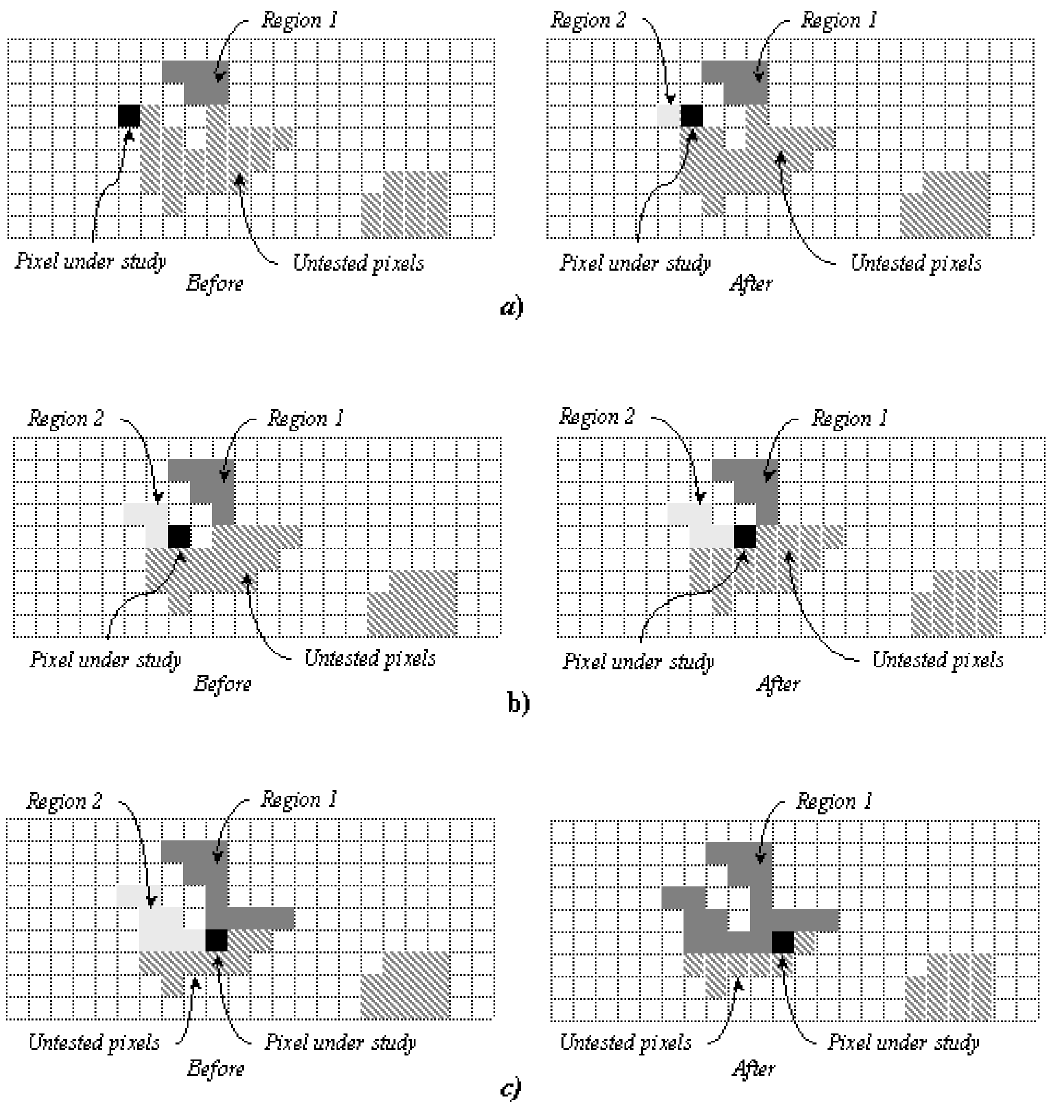


Fig. 4. Image segmentation and labeling.

5 Object Position Estimation

From all of the possible valid regions identified in a way described in the previous chapter, a proper number of regions with the biggest area is selected. This step maximizes the probability of correct regions being selected and not those due to noise which usually have small areas (one or two pixels most). The algorithm only selects the first few biggest areas and therefore small areas caused by noise are automatically excluded.

First, the team and the opponent team regions are investigated, the region being a probable team patch if it is classified as team color and if the positions of other team patches satisfy the following condition:

$$dist(\,region,team_i\,) > K_1 \tag{1}$$

where *dist* is Euclidean distance, *region* is the current testing region, *team*$_i$ are already chosen team regions and K_1 is a positive constant. In the case of the presented example the best results are obtained by $K_1 = \frac{3}{4}\, d_1$, where d_1 is the size of the color patch (see Fig. 2).

To find the right identification region among all which are classified as a particular identification color, the following condition must be fulfilled:

$$K_1 < dist(\,region,ident_i\,) < K_2 \tag{2}$$

where *region* is the current testing region, *ident*$_i$ are already chosen other identification regions (other robots) and positive constant K_2 in the presented example chosen as $K_2=d_2$ where d_2 is the robot size (see Fig. 2).

A table of possible pairs (Table 1) is generated from the selected team and identification regions with rows presenting the selected identification regions and columns presenting selected team regions. The entries in Table 1 are set to 1 if condition (2) is true. The correct team and identification pairs are then found following a simple procedure, where the element indices represent the correct pair:

- if the row has just one element equal to 1, or
- if the column has just one element equal to 1.

For the robots that cannot be identified by the above two conditions the row associated with the unidentified robot is investigated and between the possible team regions the one that has not been chosen yet is selected.

The same procedure is repeated for opponent players. The complete procedure is graphically explained in Fig. 5 where the team patches are shaded and marked with different letters, while the identification patches are marked with numbers.

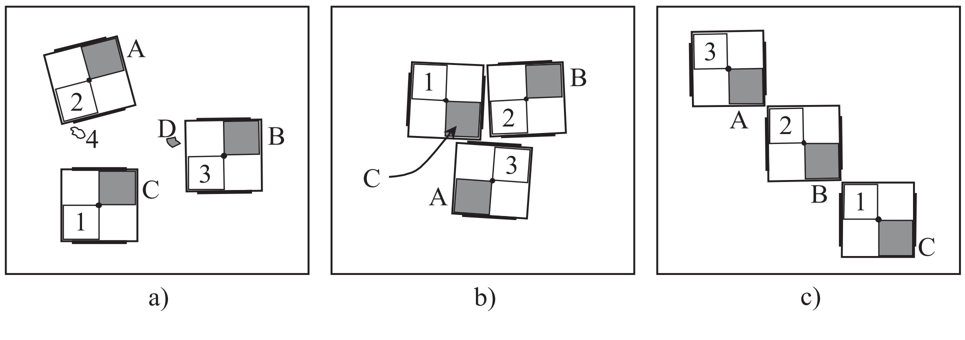


Fig. 5. Different robots placements.

For the sake of clearness only the combinations of three robots is shown in Fig. 5, nevertheless the same procedure holds for more robots also.

Table 1. Tables of possible pairs from Fig. 5.

a)				b)				c)			
	A	B	C		A	B	C		A	B	C
1			1	1			1	1		1	1
2	1			2		1	1	2	1	1	
3		1		3	1		1	3	1		

In the first table [Table 1(a)] three pairs can easily be found (1C, 2A and 3B). Regions 4 and D (see Fig. 5) are not considered because their area is small and their existence is probably due to noise. In the second table [Table 1 (b)], taking into consideration the first row and the first two columns, all pairs are found (1C, 2B and 3A). In the last table [Table 1 (c)], taking the last row and the last column, two pairs are found (1C and 3A). The second robot could not be found, thus leaving B as the only possible team color not chosen yet in the second row, the correct pair being 2B.

When all the right regions representing the color patches are found, the final estimated patch position can be improved by taking the weighted average of all region positions with the same classified color and less than distance d_1 away.

From the known positions of the regions belonging to the objects, the object positions and orientations are calculated. The position of the ball is equal to its region position, while the i-th robot data (position x_i, y_i and orientation φ_i) are calculated as follows:

$$\begin{bmatrix} x_i \\ y_i \\ \varphi_i \end{bmatrix} = \begin{bmatrix} \frac{x_{T_i}+x_{I_i}}{2} & \frac{y_{T_i}+y_{I_i}}{2} & arctan2\left(\frac{y_{T_i}-y_{I_i}}{x_{T_i}-x_{I_i}}\right) - \pi/4 \end{bmatrix}^T \tag{3}$$

with x_{Ti}, y_{Ti} denoting i-th center position of the team patch and x_{Ii}, y_{Ii} denoting i-th center position of the identification patch. Calculated orientation φ_i points to the direction of robot front side (the side next to team color, see Fig. 2).

6 Data Filtering, Camera Calibration and Non-uniform Illumination Correction

To improve the results of the presented vision system when working under hard operating conditions data filtering, camera calibration and non-uniform illumination correction algorithm should be implemented. Each of them contributes to the vision system performance if implemented in an appropriate manner. Data filtering reduces noise in estimated position data. Camera calibration corrects distortion caused by camera lens, thus enabling a more accurate and precise estimated objects positions achievement, while non-uniform illumination correction improves robustness to irregular illumination and unequal non-uniform illumination conditions.

6.1 Data Filtering

Current estimated data (positions and orientation) are usually contaminated by (camera and frame-grabber) noise and with other disturbances that appear during robot control, such as: wireless communication, wheel sliding and others. Noise level of the estimated data mostly depends on vision system initialization (determination of proper thresholds for certain patches). At proper process initialization the estimated data do not differentiate from the real data more than ±1 *mm* for positions in ±1° for orientations. Furthermore, the estimated data contains approximately one sample delayed information because of image processing. All these problems can be efficiently solved by the use of the recursive Kalman filter.

6.1.1 Data Processing by Kalman Filter

General nonlinear system can be expressed by

$$\begin{aligned}\mathbf{x}(k+1) &= f(\mathbf{x}(k),\mathbf{u}(k),\mathbf{v}(k))\\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k)+\mathbf{n}(k)\end{aligned} \tag{4}$$

where vector ***v**(k*) contains the statistically independent input noise, **n**(*k*) contains the statistically independent output noise and **C** is the output matrix. Supposing noise has zero mean value and diagonal noise covariance matrix **V** and **N**. The recursive algorithm of the extended Kalman filter [18] can be written by the prediction part (5) and correction part (6):

$$\begin{aligned}\mathbf{x}^*(k+1) &= f(\hat{\mathbf{x}}(k),\mathbf{u}(k))\\ \mathbf{P}^*(k+1) &= \mathbf{A}(k)\hat{\mathbf{P}}(k)\mathbf{A}(k)^T+\mathbf{F}(k)\mathbf{V}\mathbf{F}(k)^T\end{aligned} \tag{5}$$

where **F**(*k*) is input noise matrix. In the correction part the correction matrix **K**(*k*) is determined from previous step $\mathbf{x}^*(k)$ and from prediction value of covariance matrix $\mathbf{P}^*(k)$ of error vector. Then state estimation vector of $\hat{\mathbf{x}}(k)$ is determined and covariance matrix of error vector $\hat{\mathbf{P}}(k)$ from its prediction $\mathbf{P}^*(k)$ is calculated.

$$\begin{aligned}\mathbf{K}(k) &= \mathbf{P}^*(k)\mathbf{C}^T\left[\mathbf{C}\mathbf{P}^*(k)\mathbf{C}^T+\mathbf{N}\right]^{-1}\\ \hat{\mathbf{x}}(k) &= \mathbf{x}^*(k)+\mathbf{K}(k)\left[\mathbf{y}(k)-\mathbf{C}\mathbf{x}^*(k)\right]\\ \hat{\mathbf{P}}(k) &= \mathbf{P}^*(k)-\mathbf{K}(k)\mathbf{C}\mathbf{P}^*(k)\end{aligned} \tag{6}$$

The filter is initialized by the initial state values and a covariance matrix of the error vector. Vector $\hat{\mathbf{x}}(k)$ thus represents the filtered states of the used system and $\mathbf{x}^*(k+1)$ is a one step ahead prediction calculated from the filtered states $\hat{\mathbf{x}}(k)$.

For the robot with the differential drive from Fig. 6 kinematics equations are as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} \cos\varphi & 0 \\ \sin\varphi & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{7}$$

where v is the linear velocity, ω is the angular velocity, x and y are the position coordinates and φ is robot orientation. The discrete motion model equations obtained from (7) read:

$$\begin{aligned} x(k+1) &= x(k) + T_S v(k)\cos(\varphi(k) + T_S/2 \cdot \omega(k)) \\ y(k+1) &= y(k) + T_S v(k)\sin(\varphi(k) + T_S/2 \cdot \omega(k)) \\ \varphi(k+1) &= \varphi(k) + T_S\omega(k) \end{aligned} \tag{8}$$

where T_S is the sampling time.

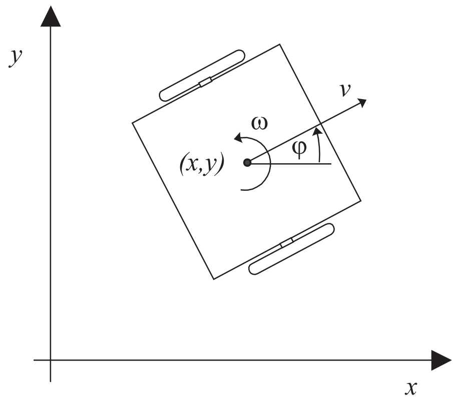


Fig. 6. Robot Architecture.

Linearization of motion equations (8) around current state estimate and input values gives:

$$\mathbf{A}(k) = \frac{\partial f(\cdot)}{\partial \hat{\mathbf{x}}} = \begin{bmatrix} 1 & 0 & -T_S v(k)\sin(\varphi(k) + T_S/2 \cdot \omega(k)) \\ 0 & 1 & T_S v(k)\cos(\varphi(k) + T_S/2 \cdot \omega(k)) \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{F}(k) = \frac{\partial f(\cdot)}{\partial \mathbf{u}} = \begin{bmatrix} T_S\cos(\varphi(k) + \frac{T_S}{2}\omega(k)) & -\frac{T_S{}^2}{2}v(k)\sin(\varphi(k) + \frac{T_S}{2}\omega(k)) \\ T_S\sin(\varphi(k) + \frac{T_S}{2}\omega(k)) & \frac{T_S{}^2}{2}v(k)\cos(\varphi(k) + \frac{T_S}{2}\omega(k)) \\ 0 & T_S \end{bmatrix} \tag{9}$$

where $\mathbf{F}(k)$ equals the input matrix of the system as noise ***v(k)*** enters the system at its inputs.

A similar procedure is performed for the ball as well. Supposing simple ball motion equation (considering dumping coefficient only)

$$\ddot{x} = -\dot{x} \cdot \lambda \tag{10}$$

and similar is done for dimension y. In discrete form at sampling time T_S the motion equation is:

$$\begin{bmatrix} x(k+1) \\ \dot{x}(k+1) \\ y(k+1) \\ \dot{y}(k+1) \end{bmatrix} = \begin{bmatrix} 1 & g \cdot T_S & 0 & 0 \\ 0 & g & 0 & 0 \\ 0 & 0 & 1 & g \cdot T_S \\ 0 & 0 & 0 & g \end{bmatrix} \cdot \begin{bmatrix} x(k) \\ \dot{x}(k) \\ y(k) \\ \dot{y}(k) \end{bmatrix} = A \cdot \mathbf{x}(k) \tag{11}$$

where $g = 1/(1 + T_S \cdot \lambda)$.

Kalman filter includes the system motion model in its filtering algorithm. However when having a larger disagreement between the used motion model and the reality, the filtered states differ from the real system ones. In the case of the robot soccer set-up this could occasionally happen during objects collisions between robots or the ball. When such cases occur, the best solution is to initialize the filter with the current state values which enables a faster recovery of the filter. Unfortunately, there are also some phenomena which cannot be reliably or even approximately predicted, such as: wheel sliding on different surfaces (clean, unclean, rough, smooth), battery condition, and the like.

It is obvious that sources of noise and other mistakes in computer vision are better suppressed (better conditions, light, system initialization, etc.) than trying to correct their consequences afterwards (noise filtering, different correction algorithms, etc.). Therefore in the sequel methods for improving the vision system performances are shown. They decreases the noise level and increase the data accuracy.

6.2 Camera Distortion Calibration

Color (and also monochrome) cameras are often used for the position estimation or tracking of mobile robots. Without accurate position estimation capabilities (e.g. navigational capabilities), mobile robots cannot be assured of having effective movement and obstacle-avoidance behavior. Whenever a good position estimation based on the acquired images is needed, the relationship between the pixel coordinates of the image and the real scene has to be known. This relationship also includes distortions caused by camera lenses. Several types of lens distortion exist; radial distortion is the most problematic one [11], especially when the inexpensive wide-angle lenses are used. The problem of camera calibration has therefore received wide attention in computer vision applications. The most frequently used method is the polynomial model for camera distortion [14, 9]. Perš *et al.* [11] suggested a mathematical model of radial distortion based on camera and lens projection geometry. Their idea is followed in the approach below.

The camera calibration routine consists of two steps originating from the demand for an easy-to-use and fully automated system, and from the fact that there are two main reasons for camera distortion. The first is radial distortion caused by the zoom (lenses); the second is perspective distortion caused by the tilt of the camera. The effects of both are illustrated in Fig. 7.

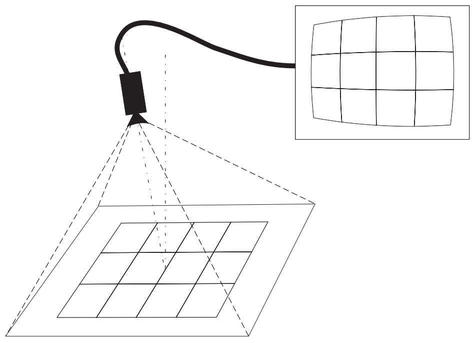


Fig. 7. Principle of camera lens distortion.

6.2.1 Radial Distortion Rectification

To correct radial distortion, a corresponding model [11] based on camera and lens projection geometry is used

$$R = f(r) = \frac{H}{2}\frac{(1-e^{-\frac{2r}{H}})}{e^{-\frac{r}{H}}} = H \cdot sinh(\frac{r}{H}), \tag{12}$$

where R is the rectified radius, r is the radius from the distorted image (Equation (13)) and H is the focal length. Let x and y be the coordinates of a pixel in the distorted image, and let X and Y be the coordinates of the same pixel in the rectified image. The origin of the transformation (12) is placed in the center of the image. The relations between (x, y) and (X, Y) are as follows

$$\begin{cases} r = \sqrt{x^2 + y^2}, \quad \varphi = arctan2\left(y/x\right). \\ X = R cos(\varphi), \quad Y = R sin(\varphi) \end{cases} \tag{13}$$

The only unknown parameter is the focal length H, which can be set manually or automatically. Since the shape of the environment, its limitations and dimensions are often known (straight parallel and perpendicular lines in robot soccer), this knowledge can be used to estimate parameter H in the model (12). If a straight parallel and/or perpendicular lines exist in the environment (e.g. boundaries, markings, etc.), the Hough transform should be used first to fit a line to the horizontal playground boundary (the longest one) on the distorted image obtained from the camera. Three points should then be selected on the line: two at the ends and one in the middle. By moving in a normal direction from the line, the pixels on the thresholded image belonging to the boundary that are closest to the previously selected points are chosen. These three pixels are then transformed by means of transformation (12) in a suitable optimization procedure to change parameter H until the most satisfactory co-linearity of the transformed pixels is obtained.

6.2.2 Rectifying the Effect of a Tilted Camera

The camera cannot always be placed over the center of the scene under observation. Its symmetrical direction is therefore not in the normal direction of the scene under observation. With an assumption of perfect projection, e.g. with a pin-hole camera, a set of parallel lines in the scene is projected onto a set of lines in the image that meet at a common point. This point of intersection, perhaps at infinity, is called the vanishing point. The three orthogonal vanishing points form a triangle, and the intersection of the triangle's heights is called the principal point.

Many precise and robust algorithms are available for the detection of perspective [12,15]. An efficient and robust method of vanishing point detection and transformation that relies only on the information gained from the playground is selected. Its basic idea was introduced by Fangi et al. [4]

$$\begin{aligned}(pX-1)x+qXy&=-X\\ pYx+(qY-1)y&=-Y\end{aligned} \tag{14}$$

where *x* and *y* are the coordinates of the rectified image, whereas *X* and *Y* are the coordinates of an image with rectified radial distortion only. Parameters *p* and *q* are estimated from the location of vanishing points, which are defined by the cross-sections of the lines fitted (by means of Hough transformation) to the playground boundaries of an image with rectified radial distortion. The principal point is placed in the center of the image. The two lines running from the principal point to the calculated vanishing points are almost perpendicular. From the position of the vanishing points and the principal points, playground rotation is determined. If the detected rotation is not negligible, the rotation should also be rectified prior to perspective transformation. Parameters *p* and *q* from Equation (14) are obtained as

$$p = 1/d_{V1}, \quad q = 1/d_{V2} \tag{15}$$

where d_{V1} is the distance from the principal point to the horizontal and d_{V2} is the distance to the vertical vanishing point.

Although both radial distortion and perspective correction are very simple, running the algorithm on every incoming image from the camera is time-consuming. Pixel classification, image segmentation and other vision-based algorithms are therefore better performed on a distorted image and only the estimated objects coordinates are transformed (radial distortion and perspective rectification).

6.3 Non-uniform Illumination Compensation

Good illumination conditions contribute to the efficiency of every computer vision application. However, uniform illumination is usually difficult to obtain; vision algorithms must therefore somehow be adapted to such circumstances (especially in color-based vision tracking). Non-uniform illumination caused by irregular illumination is the main reason for the bad tracking or even for the loss of position estimation of mobile robots. If a vision system is taught a specific color when the carrier is somewhere in the center of the observed scene, there is a strong possibility that this color will not be recognized when its carrier is positioned in one of the corners. If non-uniform illumination and illumination conditions do not change drastically during operation, a static shade model could be built in the initialization phase. The relation between the true (ideal uniform illumination) image $U(x,y)$ and acquired image $N(x,y)$ is usually described by means of a linear model of image formation

$$N(x,y) = U(x,y)S_M(x,y) - S_A(x,y) \tag{16}$$

where $S_M(x,y)$ is the multiplicative and $S_A(x,y)$ the additive component. Retrospective approaches and acquisition-based ones are two different types of possible non-uniform illumination correction method. Acquisition-based methods involve taking one or two reference images and interpreting them as multiplicative and additive components for the acquired image, while retrospective methods rely solely on the information content of the acquired image. The most intuitive retrospective methods for correcting multiplicative and smooth intensity variations are homomorphic filtering, image blurring, smoothing, averaging, Fourier-domain filtering and homomorphic unsharp masking [7]. In [6], additive and multiplicative non-uniform illumination components were approximated by means of second-order polynomials. Our test showed that this type of model did not improve detection, and sometimes even made it worse. The reason for such results lies in the shape of the illumination plane. A typical illumination plane consists of several peaks and valleys which cannot be modeled sufficiently by means of second-order polynomials but need higher order approximations. It may be quite difficult to estimate the proper order of a polynomial in a model, hence a simpler method was chosen. The proposed method is acquisition-based; it employs a background image and the additive component is discarded. Non-uniform illumination is finally corrected by relation (17)

$$\hat{U}(x,y) = \frac{N(x,y)}{N_{BCK}(x,y)}C \tag{17}$$

where $\hat{U}(x,y)$ is the corrected image, $N_{BCK}(x,y)$ is the background image and C is the normalization constant needed to restore the desired grey level range. To improve computational time efficiency, multiplication by component $C/N_{BCK}(x,y)$ in Equation (17) should be realized by means of a look-up table.

6.3.1 Acquiring the Multiplicative Component

The camera's field of view should be covered with grey paper sheets (not necessary if the scene is uniformly colored already). The acquired reference image $N_{BCK}(x,y)$ then represents the non-uniform illumination plane of a multiplicative component $C/N_{BCK}(x,y)$ in relation (17). The reference image is retouched with a slight Gaussian blurring to increase the smoothness of the non-uniform illumination plane. The illumination planes are then calculated for all three channels in the RGB (Red, Green, and Blue) color space. Fig. 8 shows a typical illumination plane calculated from the robot soccer court in Fig. 11 for the red channel only, while the other two have a similar shape.

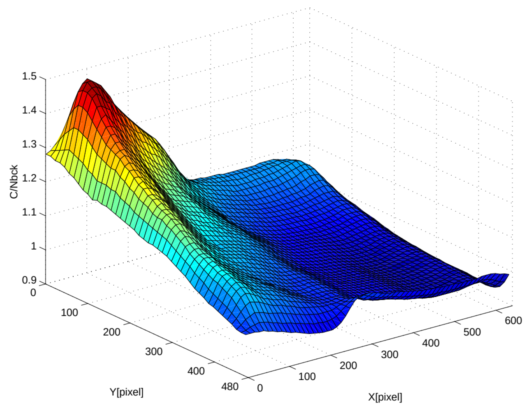


Fig. 8. Calculated multiplicative component (C/N_{BCK}) for the red channel.

Among a number of available color spaces, the RGB color space is usually output in most color cameras and frame-grabbers and as such easy to implement in computer systems, television, video, etc., while in image processing the HSI and YUV color spaces seem more appropriate when there are different illumination conditions. They code the information about chrominance in two dimensions (H and S, or U and V), with only one dimension including information about intensity (I or Y).

In our experiments, both RGB and YUV color spaces were tested on incoming camera images. The classification results obtained were similar. However, to obtain the YUV color representation, transformation from the original RGB space had to be performed. This transformation was time-consuming, although optimization by means of look-up tables was used. Including optimization, it requires some 30 ms, while the rest of the program takes only 8 ms to identify objects from the image. The RGB color space is therefore used in the experiments presented. The illumination plane from Fig. 8 is then applied to incoming camera images (in accordance with equation (17)).

7 Application of Proposed Corrections to the Robot Soccer Game

The suggested approaches for the data filtering, rectification of camera distortions and non-uniform illumination are demonstrated on a robot soccer set-up (Fig. 1). Each of them improves the results of the vision system if implemented in the appropriate manner. Evaluations of the improvements are also given.

7.1 Data Filtering

The implementation of Kalman filter is validated on a real robot from Fig. 6. First, noise variances were determined from an observing standing robot. The results from the above-presented filter for the robot can be seen in Fig. 9 where camera estimated data and filtered data are shown.

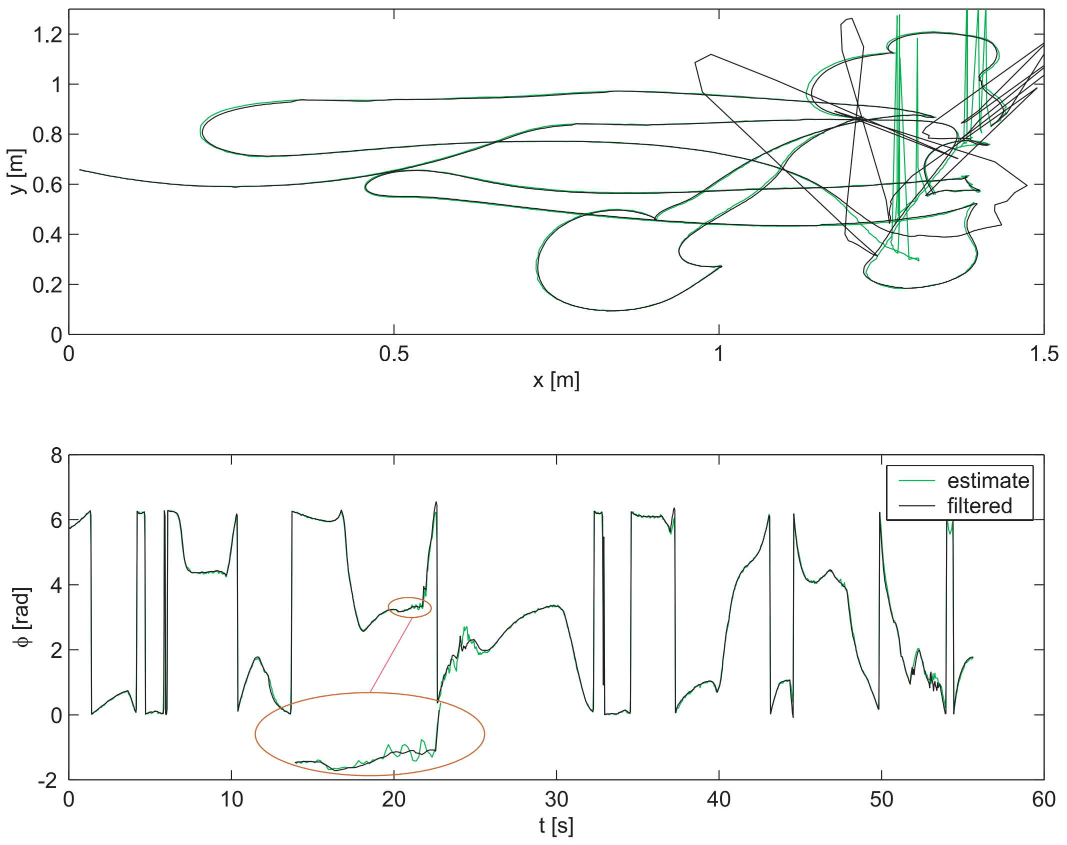


Fig. 9. Robot trajectory and orientation; real and filtered data.

It can be concluded that orientation data are more subjected to noise than position data. Therefore Kalman filter proved useful especially at orientation estimation. An important advantage is also prediction of the robot states in future or at least for the delay time resulting from image processing. The used model also does not include collisions among robots or between robots and boundary. Therefore in such cases the estimated data values could vary from real data. As already mentioned the best strategy and solution to collision situations is to initialize a filter with the current robot state values enabling a faster recovery of the filter. Another advantage of the filter is also its ability to filter out major distinctions (vertical spikes on upper graph in Fig. 9) resulting from bad illumination conditions or vision system initialization. In these situations the robot data are wrongly estimated. On filtered data this is hardly observed as such behavior is unlikely (according to robot model) and is successfully filtered out.

Filter validation of the ball motion across the playground gives similar results and can be observed in Fig. 10.

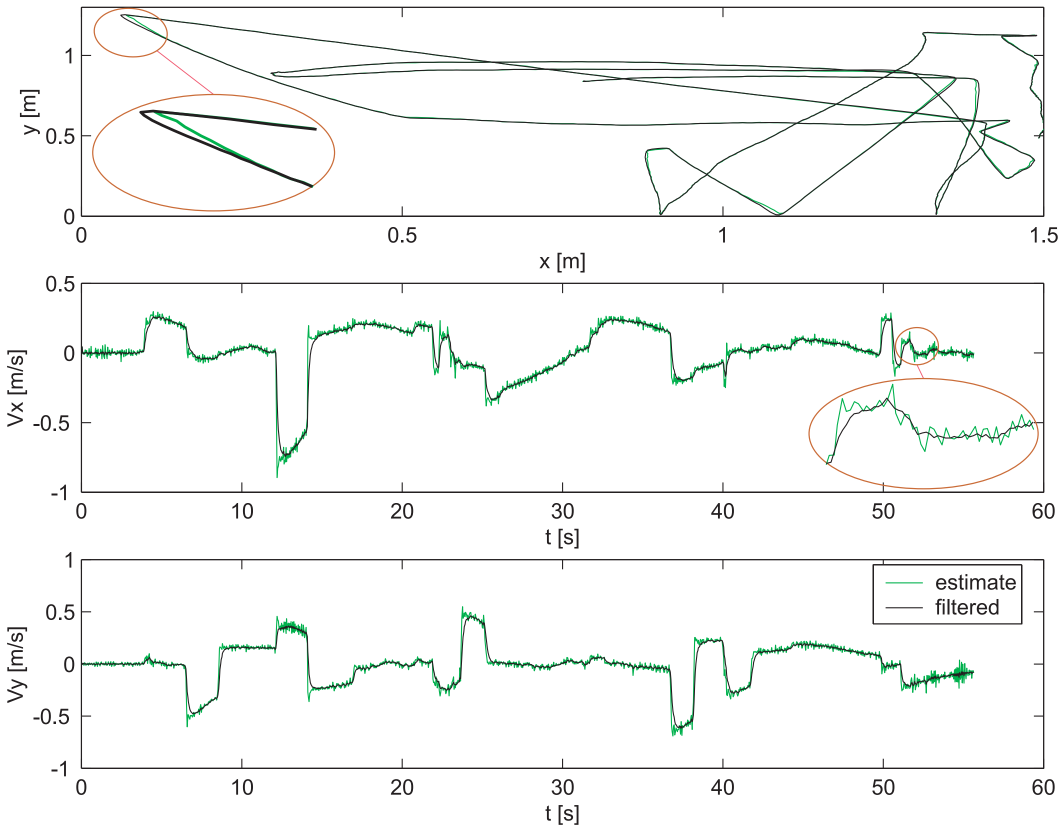


Fig. 10. Ball path and velocity in both directions *x* and *y*.

Ball velocities are determined by differentiation of ball position data. Details from upper and middle graphs in Fig. 10 show a typical consequence of a ball collision with some object which should be treated filter initialization.

7.2 Camera Distortion Correction

In a robot soccer game, each team has its own camera. Each of them is placed slightly to the left or right of the center of the playground. In order to cover as much of the acquired image of the playground as possible, the camera has to be tilted, which causes perspective distortion. The camera must also see the whole playground. According to the game rules it must not be higher than 2 meters above the playground. Therefore wide-angle lenses are used; these cause radial distortion.

An example of the complete camera calibrating procedure is shown in the figures below. The distorted image obtained from the camera is shown in Fig. 11, while the rectified image can be seen in Fig. 12.

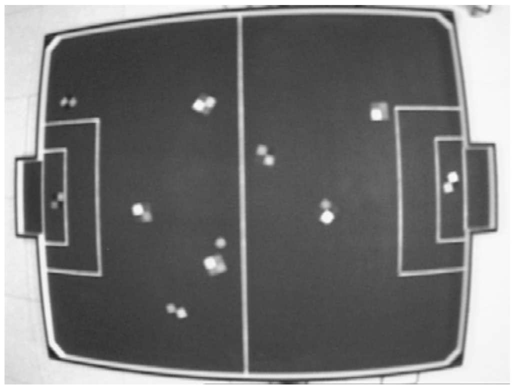
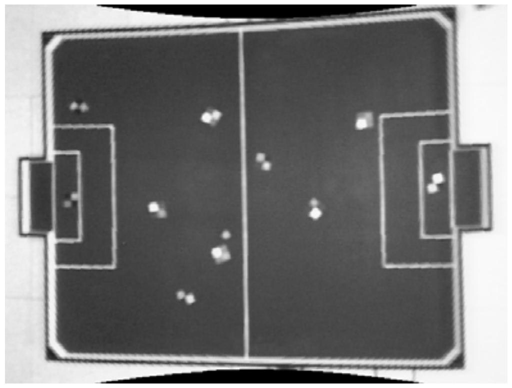
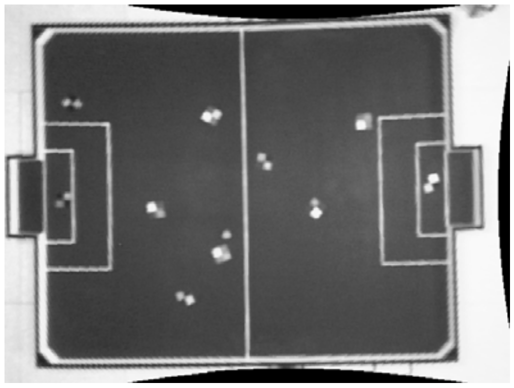


Fig. 11. Distorted image (image from the camera with radial and perspective distortion).



a)



b)

Fig. 12. Rectified image: (a) corrected radial distortion only; (b) corrected radial and perspective distortion.

7.3 Non-uniform Illumination Compensation

Although the playground is illuminated by a number of different light sources (reflectors or neon tubes), it is practically impossible to obtain uniform illumination. If lights are arranged carefully, compensation may not be needed, but this is usually not the case.

To test the efficiency of the non-uniform illumination correction algorithm, a number of color patches of the same color are arranged over the playground (see Fig. 13). From the color patch in the center of Fig. 13, the representative color thresholds are determined. The pixel classification and segmentation algorithm is first represented on the incoming camera image (Fig. 14), and then on the same image with compensated non-uniform illumination (Fig. 15).

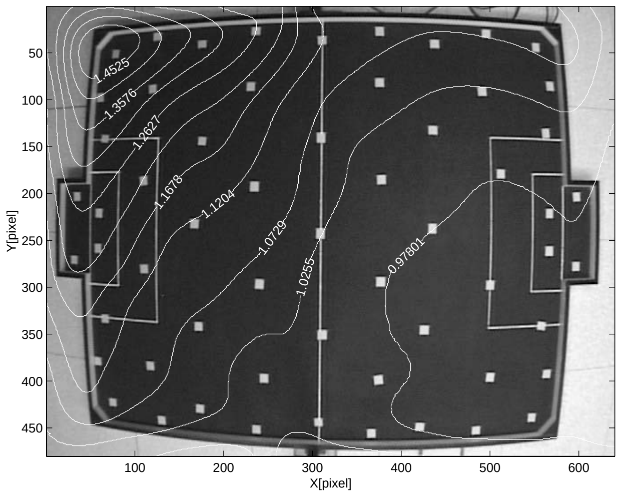


Fig. 13. Camera image with color patches arranged over the playground and intensity illumination contours displayed.

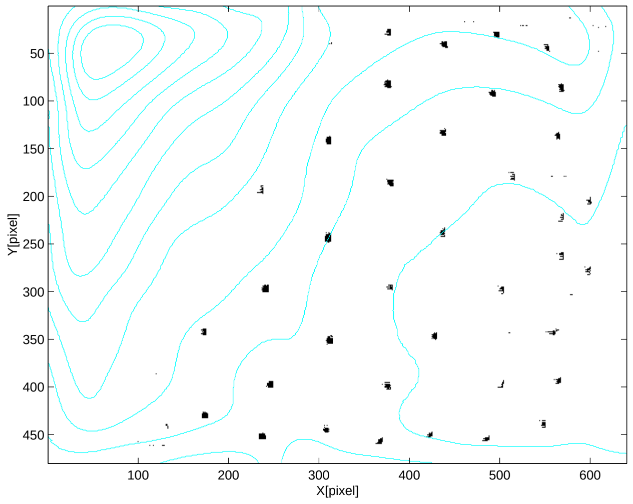


Fig. 14. Classified pixels from the camera image in Fig. 13.

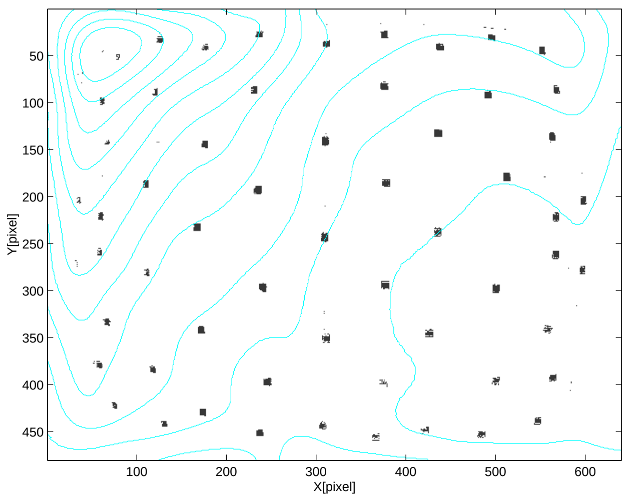


Fig. 15. Classified pixels from the camera image in Fig. 13 with compensated non-uniform illumination.

In Figs. 14 and 15, the regions with black color represent recognized pixels which belong to color patches from Fig. 13. A pixel is classified as belonging to the color patches if it fits the representative color thresholds. Due to non-uniform illumination (see the intensity illumination contours in Figs. 13-15 or the illumination plane in Fig. 8), some color patches in Fig. 14 are purely detected or even undetected.

From Fig. 15 it can be seen that non-uniform illumination compensation improves the efficiency of the classification and segmentation algorithm. All the patches from Fig. 13 can be reliably identified from the compensated image (Fig. 15); this could not be achieved by using the non-compensated image (Fig. 14).

7.4 Evaluation of Results

The suggested approaches for the estimated data filtering, rectification of camera distortions and non-uniform illumination applied to a robot soccer set-up improve the game in many ways.

As already mentioned, at corresponding vision system initializations and good lightening conditions the noise level does not contaminate estimated data more than ±1 mm for positions in ±1° for orientations (evaluations taken when robots stand still). However, noise level increases at worse conditions. The data filtering presented is useful because it suppresses noise and other disturbances and eliminates delays resulting from image processing by predicting future robot states. The obtained filtered data therefore enables a better (t.i. more accurate and faster) robot control.

Radial and perspective distortions are successfully corrected, which can be proved by obtaining straight parallel and perpendicular lines for playground boundaries (see Fig. 12) from the distorted image (Fig. 11) of the playground. More accurate and precise estimations of robot positions are therefore achieved, which is especially important for robot control when robots are close to boundaries. The improvement is difficult to estimate because this depends on where the robots are mostly moving during the game (at the corners the distortions are higher than in the center). If they are mostly moving in the corners of the playground, the precision of the robot's positions and the quality of the game (boundary avoidance, ball manipulation) is improved by up to 40 percent (statistics taken from our game).

Non-uniform illumination compensation improves robustness to irregular illumination. Patches in certain areas on the playground which cannot be found from the distorted image are then reliably found (recognition rate of around 98 percent). Recognition rate is thus improved by up to approximately 30 percent compared to the rate for poor illumination conditions (statistics taken from our game).

8 Conclusion

The issues the chapter address are vexing ones for the robot soccer community; issues that are often passed as trivial. However, there is little literature on effective solutions to the problems, which is often source of frustration to teams who wish to purse other issues in the soccer domain (such as AI and control).

An example of establishing a fast and robust vision system for the purpose of mobile robots in soccer game is presented. Special consideration is given to optimization of computational work and robustness issues. The latter are assured by inclusion of methods for image quality improvement such as correction of nonuniform illumination and camera lens distortions. Robustness is further achieved by time-efficient algorithms which enable global image processing. Contrary, some vision systems used by other robot soccer teams employ local image processing to obtain the desired frame rate of the vision system. The major disadvantage of these algorithms is loss of one or more objects (robots or ball) because of some unpredicted reasons (lightening conditions, collisions, bugs). The local search areas have to be increased until objects are found, which results in larger and irregular sample time. This could not happen with global image processing. However, disadvantage of the presented approach can appear if a large number (more than 15) of different color patches have to be followed. Some of color patches could then become quite similar on camera image which could result in wrong objects estimation. The problem will be dealt with in the future work by inclusion of object tracking algorithms.

Further on an approach towards establishing a more robust and accurate vision system for mobile robot tracking under poor illumination and camera lens distortion conditions is presented. To improve the results of visual robot tracking, a camera calibration and non-uniform illumination correction algorithm are suggested. The former corrects the distortion caused by the camera lens, thus achieving a more accurate and precise estimation of object position, while the latter improves robustness to irregular illumination and non-uniform illumination conditions. The applicability of the suggested solutions is demonstrated in a robot soccer game, where any incorrect or inaccurately estimated robot or ball position results in poor game-play (apart from perfection of the strategy control algorithm). The robustness of the vision system is therefore improved by means of camera calibration algorithms. The suggested procedure for shading correction proved useful when the illumination conditions remained more or less unchanged during the game. The procedure presented also assumes fixed camera view, as in central vision systems. In general mobile robotics, these conditions are not always met. If illumination conditions change during tracking, a more robust approach with an adaptation mechanism should be applied. This will be addressed in further research. The optimized algorithms presented enable the vision system to be used in real-time applications where robustness to irregular illumination and camera distortions are important.

References

[1] Bishop, C.M. *Neural Networks for Pattern Recognition*; Oxford University Press: Oxford, UK, 1995.

[2] Bradski, G.R. Computer Vision Face Tracking For Use in a Perceptual User Interface, *Intel Technology Journal*. 1998, 2nd quarter, 1-15.

[3] Bruce, J.; Balch, T.; Veloso, M. Fast and inexpensive color image segmentation for interactive robots, in: *Proceedings of IROS-2000*, Takamatsu, Japan, October 2000, pp. 2061-2066.

[4] Fangi, G.; Gagliardini, G.; Malinverni, E.S. Photointerpretation and small scale stereoplotting with digitally rectified photographs with geometrical constraints, in: *CIPA congres 2001*, 18-21 September 2001. http://cipa.icomos.org/papers/2001-.htm.

[5] Klančar, G.; Orqueda, O.; Matko, D.; Karba, R.; Robust and efficient vision system for mobile robots control - application to soccer robots, *Electrotechnical Review, Journal for Electrical Engineering and Computer Science*. 2001, Vol. 68, No. 5, 306-312.

[6] Likar, B.; Maintz, B.A.; Viergever, M.A.; Pernuš, F. Retrospective shading correction based on entropy minimization, *Journal of Microscopy*. 2000, Vol. 197, 285-295.

[7] Likar, B.; Viergever, M.A.; Pernuš, F. Retrospective correction of MR intensity inhomogeneity by information minimization, in Lecture Notes in Computer Science; Delp, S.L.; DiGioia, A.M.; Jaramaz, B.; Eds.; *Medical Image Computing and Computer-Assisted Intervention*; Springer: Berlin, GER, 2000, Vol. 1935, pp. 375-384.

[8] Looney, C.G.; Pattern Recognition using Neural Networks; *Theory and Algorithms for Engineers and Scientists*; Oxford University Press: UK, 1997.

[9] Pajdla, T.; Werner, T.; Hlavač, V. (1997). Correcting Radial Lens Distortion without Knowleadge of 3-D Structure, *Technical report TR97-138*, Faculty of Electrical Engineering, Czech Technical University, Praha, Czech Republic.

[10] Pavešić, N.; Razpoznavanje vzorcev: Uvod v analizo in razumevanje vidnih in slušnih signalov (Pattern Recognition*; An Introduction into the Analysis and Understanding of Visual and Audio Signals)*, Faculty of Electrical Engineering: Ljubljana, Slovenia, 2000.

[11] Perš, J.; Kovačič, S. Nonparametric, Model-Based Radial Lens Distortion Correction Using Titled Camera Assumption, in *Proceedings of the Computer Vision Winter Workshop 2002*; Wildenauer, H.; Kropatsch, W.; Eds.; Bad Aussee, Austria, 2002; pp. 286-295.

[12] Rother, C. A new Approach for Vanishing Point Detection in Architectural Environments, in: *Proceedings of the 11th British Machine Vision Conference (BMVC'00)*; Bristol, UK, 2000; pp. 382-391.

[13] Sargent, R.; Bailey, B.; Witty, C.; Wright, A. The importance of fast vision in winning the First Micro-Robot World Cup Soccer Tournament, *Robotics and Autonomous Systems*. 1997, Vol. 21, 139-147.

[14] Tsai, R.Y. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses, *IEEE Journal of Robotics and Automation.* 1987, Vol. 3, No. 4, 323-344.

[15] Tuytelaars, T.; Proesmans, M.; Gool, L.V. The cascaded Hough transform, in: *Proceedings of International Conference on Image Processing (ICIP '97)*; Washington, DC, 1997; Vol. 2, pp. 736-739.

[16] Wyeth, G.F.; Brown, B. Robust Adaptive Vision for Robot Soccer, in: *Mechatronics and Machine Vision in Practice*; Billingsley, J.; Eds.; Research Studies Press, 2000; pp. 41-48.

[17] Rosenfeld, A., Pfaltz, J. L. Sequential Operations in Digital Picture Processing, *Journal of the ACM*. 1966, Vol. 3, No. 4, 471-494.

[18] Jetto, L.; Longhi, S.; Venturini, G. Development and Experimental Validation of an Adaptive Extended Kalman Filter for the Localization of Mobile Robots, *IEEE Transactions on Robotics and Automation*. 1999, Vol. 15, No. 2, pp. 219-229.