ELSEVIER

Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Robotics and Autonomous Systems xxx (2004) xxx–xxx

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

Color-based visual servoing under varying illumination conditions

Geert De Cubber*, Sid Ahmed Berrabah, Hichem Sahli

Vrije Universiteit Brussel (VUB), Pleinlaan 2, B-1050 Brussels, Belgium

Received 30 July 2003; accepted 8 March 2004

Abstract

Visual servoing, or the control of motion on the basis of image analysis in a closed loop, is more and more recognized as an important tool in modern robotics. Here, we present a new model-driven approach to derive a description of the motion of a target object. This method can be subdivided into an illumination invariant target detection stage and a servoing process which uses an adaptive Kalman filter to update the model of the non-linear system. This technique can be applied to any pan–tilt zoom camera mounted on a mobile vehicle as well as to a static camera tracking moving environmental features.

Keywords: Color constancy; Visual servoing; Target tracking; Bayesian modeling

1. Introduction

The implementation of a system capable of performing visual servoing in everyday environments requires careful consideration of the mechanical, control and vision issues involved in the closed-loop sensing system. The primary elements are the detection of objects of interest moving in the scene and their subsequent more detailed analysis during tracking over time. Mechanically, this requires a pan–tilt camera platform. The visual servoing approach is based on an information feedback loop, which determines an error vector defined in the vision space. This vector is updated after every image acquisition. In a target-tracking scheme, the error vector is defined as a measure, at a given time, of the distance in image coordinates between the target position and the image center. This error serves to determine the control parameters of the pan–tilt platform (camera).

The scheme proposed here, consists of a two-phase process, where the first phase deals with target detection. In the proposed approach, the target is distinguished from the environment based upon its color value. One of the major problems arising here is the effect of an ever-changing illumination, as a change in illumination will also change the perceived colors—or more generally the perceived image—of the environment. To counter this, a color constancy approach is presented to improve the classification capabilities of the color target-tracking algorithm. Color constancy, as defined in [20], is the ability to recover a surface description of color, independent

* Corresponding author. Tel.: +32-26292982; fax: +32-26292883.
E-mail addresses: gdcubber@etro.vub.ac.be (G.D. Cubber), saberrab@etro.vub.ac.be (S.A. Berrabah), hsahli@etro.vub.ac.be (H. Sahli).

0921-8890/$ – see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.robot.2004.03.015

of the illumination. The applied approach consists of building up a reliable model to retrieve the reflection characteristics of the object to be tracked, while eliminating as much as possible interfering effects due to illumination changes, shadows, specular reflections, etc. A Bayesian framework is used to build and update this model over time.

In the second phase, the one of the visual servoing, the motion model of the target object is retrieved. This movement is not known a priori and the perspective projection relationship is a non-linear one, so the servomotor–camera–target system is non-linear and time-variant. This system can be approximated as a linear time-variant one, such that an observer-based full-state feedback control can be used to implement the tracking function. From this online identification process, the system-modeling problem is solved. The simplified linear model is used to approximate the more complicated system, while the method of the observer-based full-state feedback control guarantees the system stability. The parameters for the control of the camera can be estimated by considering the position of the detected target in the image plane and its evolution in time. To make the visual control loop compatible with the real-time constraint, a windowing technique is used for the image processing task, such that only a small window around the detected object is processed. An Extended Kalman Filter is used to predict the future size and position of the window in the image plane, while the target is moving in 3D space.

1.1. Previous work

This article focusses on two distinct research topics: color constancy and visual servoing and how they can be combined. Several research works have been shown in both of these areas.

In the field of color constancy, the first computational model was proposed by Land and McCaan [18]. Their retinex theory assumes a Mondrian world, which consists of planar patches of differently colored paper. The illumination across this Mondrian world is assumed to be smoothly varying over the observed scene. In this setup, sharp changes in color signal intensity can be attributed to object boundaries, whereas smooth changes are due to illumination variation. In general, the algorithm can determine constant color descriptors despite changes in illumination. However, if the scene surrounding a patch is changed, different color descriptors are generated.

By far the simplest color constancy method is the gray world algorithm. It goes out from the assumption that the average of all colors in an image is gray, so the red, green and blue components of the average color are equal. The amount the image average departs from gray determines the illuminant *RGB*.

Another widespread approach is the white patch algorithm, which is at the heart of many of the various Retinex algorithms. It presumes that in every image there will be some surface or surfaces such that there will be a point or points of maximal reflectance for each of the R, G, and B bands.

A more sophisticated solution is presented by the gamut constraint method. The fundamental observation of this method is that not all possible *RGB* values will actually arise in images of real scenes. The convex hull of the set of *RGB* values of a certain surface obtained under the canonical illuminant is called the canonical gamut. When using the gamut constraint method, the color constancy problem is brought down to find the transformations mapping the *RGB* values under new illuminants to the canonical gamut.

Most modern approaches to color constancy use a finite-dimensional linear model in which surface reflectance and illumination are both expressed as a weighted sum of fixed basis functions [2,10,16,23]. The task of color constancy, therefore, becomes that of estimating the reflectivity weights for the object and the illumination weights. Typically the scene is assumed to be Mondrian and composed of Lambertian surfaces.

The extension of color constancy to more natural scenes, with varying scene geometry and surfaces that exhibit glossy reflection, has been considered by D'Zmura and Lennie [37]. They used the dichromatic reflection model to describe interface and body reflection processes.

Recently good results have been achieved using a neural net to estimate the chromaticity of the illuminant [9]. Here a neural net is trained on synthetic images randomly generated from a database of illuminants and reflectances. The concept of color constancy has been used before in the context of object recognition. In [24], Matas et al. model objects in a test database under a range of expected illuminations. Each surface on a specific object is represented by a convex set of the possible chromaticities under the range of possible illuminations. The occurrence of a chromaticity in this range is a vote for the presence of the object. In this manner, the likelihood of the presence of each object can be estimated.

In his Ph.D. work Barnard [1] studies the performance of different color constancy algorithms. He concludes that the errors remain considerable even for the most performing algorithms under laboratory conditions. These techniques also typically require hours of calculation time to process one non-synthetic image, making them totally unfit for real-time and real-world vision tasks.

In the present work, a color constancy technique is proposed for real-time target identification under varying illumination conditions. A finite-dimensional linear model is built up using Bayesian reasoning.

In the field of visual servoing, the research is even more extended and is becoming more and more important with the steady increase in computing power. In the past, the complexity of the vision algorithms needed to process the acquired images, restricted real-time—and therefore also real-world—applications. A comprehensive study of research results so far can be found in [7]. In this work, Corke shows that the concept of visual servoing has known a considerable evolution since it was first introduced by Hill and Park in [15]. To clearly state the position of the present work, it is useful here to make a classification of the existing techniques.

From one point of view, one can consider the approaches where the camera is fixed at a certain point in the world coordinate system and on the other hand the eye-in-hand configuration, where the camera is fixed on the end effector or mounted on a mobile robot [35,36]. A classification can also be made by separating the monocular vision systems from the stereo vision systems. Stereo vision is better suited to retrieve the much needed 3D-data out of the environment, but on the other, it is more expensive and adds to the complexity of the general system, thereby making real-time performance more difficult. A distinction needs also to be made between model-based and model-free or model-independent approaches. Whereas most researchers nowadays choose to build up some kind of dynamic 3D model of the target [4], others [27] have shown good results with model-independent approaches.

Another important classification was made by Sanderson and Weiss in [29], where they marked the difference between image-based and position-based servoing. Other authors refer to these concepts respectively as 2D and 3D visual servoing [8,21]. In a position-based control scheme, the control is directly based upon the error on the position of the camera. To estimate this error, image features are extracted and then the pose of the target can be calculated through the knowledge of a geometric model of the target. This process involves inverse kinematics which requires generally a very accurate kinematic model of the robot–camera—or more general target–camera—system. Small errors in the model, measurements, or camera calibration can lead to a servoing failure. Another disadvantage of the position-based approach is the need for a considerable amount of a priori knowledge. As an advantage, the position-based control scheme performs a target positioning by definition and can therefore directly control the camera trajectory in Cartesian space. Position-based visual servoing has been applied mainly to robot-arm manipulators, where the kinematic model is well known and often by using stereo vision systems [11,34]. When using an image-based servoing scheme, the control error function is expressed directly in the 2D image space. This allows for faster tracking, yet it poses a difficult task to the controller since the process will generally be non-linear, highly coupled and time-variant. A whole variety of image-based visual servoing approaches have been shown [3,19,28], where the research is generally mainly focussed at the design of the controller. It should be noted that other options exist besides position-based and image-based visual servoing. A less common technique is for example the motion-based approach, which employs the optical flow for tracking [26].

In the present work, a visual servoing approach is proposed which uses a monocular vision system. This work tries to integrate the benefits of position-based and image-based servoing by incorporating an online identification method to estimate the dynamic system model of the target to control the camera. This model is used in a Kalman filter for tracking. The algorithm is also capable of estimating the 3D-coordinates of the target object in a separate process. This means that the presented system is capable of delivering the same data (3D-localization) as a position-based approach, while avoiding the exact knowledge of the kinematic model.

2. Illumination invariant classification

2.1. Modelization

2.1.1. The color reflection model

Our approach is directly based upon the physical characteristics of color reflection. The main problem for the correct interpretation of a camera image is that the measured intensities are function of a large number of parameters and most of them cannot be retrieved in any possible way due to their strong interconnectivity. The color of an object in the image must be considered as an appearance rather than as a real material property. Nevertheless, color can be used to identify objects as long as the parameters which influence the formation of the perceived color are taken into account. To do so, we make use of the dichromatic reflection model, which was first introduced by Shafer in [30]:

$$\rho_c = k_{\mathrm{b}} \int_\lambda e(\lambda) \cdot f_c(\lambda) \cdot r_{\mathrm{b}}(\lambda)\, \mathrm{d}\lambda + k_{\mathrm{s}} \cdot \int_\lambda e(\lambda) \cdot f_c(\lambda) \cdot r_{\mathrm{s}}(\lambda)\, \mathrm{d}\lambda, \tag{1}$$

where ρ_c is the measured intensity of channel c, $e(\lambda)$ the normalized light spectrum, $f_c(\lambda)$ the cth channel sensor response function, $r_{\mathrm{b}}(\lambda)$ the body reflectance function, $r_{\mathrm{s}}(\lambda)$ the surface reflectance function, k_{b} the attenuation factor for the body reflectance and k_{s} the surface reflectance attenuation factor.

2.1.2. Color spaces

In computer vision, a color is generally represented using a triplet of intensity values. The exact meaning of each of these values is determined by the choice of color space. This choice should be made taking into account the choice for the distance operator used to calculate the color "difference" between two pixels. Among the different color spaces, our choice went out to the $l_1l_2l_3$-space, a color space which was originally introduced by Gevers and Smeulders in [12]. It poses an attractive alternative to the HSI space due to its computational simplicity. The space can be formulated as follows:

$$l_1 = \frac{|R-G|}{|R-G|+|R-B|+|G-B|}, \qquad l_2 = \frac{|R-B|}{|R-G|+|R-B|+|G-B|},$$
$$l_3 = \frac{|G-B|}{|R-G|+|R-B|+|G-B|}. \tag{2}$$

In [13], Gevers and Stokman prove that according to the dichromatic reflection theory, this space is invariant to highlights, viewing direction, surface orientation and illumination direction. This means that we can work with a simplified form of Eq. (1):

$$H_{l_1l_2l_3}(x,t) = \int_\lambda e(\lambda,t) \cdot f_c(\lambda) \cdot r_{\mathrm{b}}(\lambda,x)\, \mathrm{d}\lambda. \tag{3}$$

For the distance operator, two classical options dominate the field: Euclidean distance and vector angle. Wesolkowski concludes in [33] that the vector angle is the best overall distance operator, with the disadvantage that is ignores intensity. However, in the case of the $l_1l_2l_3$ color space, the difference is not noteworthy, so we chose for the computational simplicity of the Euclidean distance approach.

2.1.3. Discretization

Eq. (3) can be discretized by sampling over a number of wavelength bands. We chose to use a finite-dimensional linear model with a limited amount of parameters:

$$e(\lambda,t) = B_{\mathrm{e}} \cdot q_{\mathrm{e}}, \qquad r_{\mathrm{b}}(\lambda,x) = B_{\mathrm{r}} \cdot q_{\mathrm{r}}. \tag{4}$$

The columns of the $N \times N_e$ matrix B_e and those of the $N \times N_r$ matrix B_r represent the basis functions for the light and the reflectance spectrum respectively. The N_e element q_e vector and the N_r element q_r vector describe respectively the illuminant and the body reflectance spectrum. The basis functions can be obtained by applying principle component analysis on data from spectrometers. For real-time target tracking using only a simple camera, this is not an option, so this would force us to use premade sets of basis functions. Using repeated daylight measurement data, the CIE setup such a three-dimensional linear model [5], while others [17] used four-dimensional models. For the reflectance spectrum, Cohen [6] and Maloney [22] conclude that natural spectra lie within small-dimensional linear models and that four-dimensional models suffice to approximate most materials. However, this goes out from the assumption that one can retrieve high quality from the illuminant spectrum using expensive spectrometers. In general, it is wiser to work with a more extended set of basis functions when such high-quality data is not present. Our tests pointed out that three or four dimensions did not suffice (at least with the data we could retrieve) to describe the illuminant spectrum and as a result we chose to use 10 basis functions.

If $D(f_c)$ is the $N \times N$ diagonal matrix with f_c as diagonal elements, we get by inserting Eqs. (4) and (3):

$$h_c = q_e^T \cdot B_e^T \cdot D(f_c) \cdot B_r \cdot q_r. \tag{5}$$

The problem with this representation is that the basis and sensor sensitivity functions are not well known. To avoid this difficulty, we use an approach similar to the one described in [31], which introduced a lighting and reflectance matrix, parameterized using $4 \times N_e$ variables in a manner independent of basis functions and sensitivity functions. The idea is to write the vector $B_e^T \cdot D(f_c) \cdot B_r \cdot q_r$ as σ_c, which is an alternative descriptive function for the body reflectance function and which can be used to discriminate between observed materials. This leads to a general equation:

$$h^T = q_e^T \cdot \sigma, \tag{6}$$

where h^T represents the color triplet in the $l_1 l_2 l_3$ color-space and σ is an $N_e \times 3$ matrix holding all the reflection characteristics independently of the illumination. This matrix needs to be estimated and based upon this estimate the classification process can be performed.

2.2. Bayesian color classification

2.2.1. Learning

In a learning phase, the algorithm learns the reflection characteristics of the object to be tracked. Small patches of images are accumulated over time while the material in question is subjected to a varying illumination. All intensity measurements h are combined in an $f \times 3p$ color measurement matrix H, while p is the number of pixels in the scene patch and f the number of frames sampled. If we sample for long enough, then eventually f will grow larger than p and the light spectrum matrix Q and the reflection characteristics matrix S can be recovered by applying singular value decomposition on H, while $H = Q \cdot S$:

$$H = \begin{pmatrix} h(x_1,t_1)^T & \cdots & h(x_p,t_1)^T \\ \cdots & \cdots & \cdots \\ h(x_1,t_f)^T & \cdots & h(x_p,t_f)^T \end{pmatrix}, \qquad Q = \begin{bmatrix} q(t_1)^T & \cdots & q(t_f)^T \end{bmatrix}^T,$$
$$S = \begin{bmatrix} \sigma(x_1) & \cdots & \sigma(x_p) \end{bmatrix}, \tag{7}$$

$p(q_e|l)$ represents the light spectrum distribution if the illuminant l is known. It can be calculated at this moment, because Q is independent of the material. We use an Expectation Maximization (EM) clustering method to derive the reflection distributions. This algorithm applies multivariate Gaussian mixture modeling with an unknown number of mixture components, so the number of clusters is not fixed on beforehand, which makes the classification very flexible. To estimate the number of clusters or mixtures to be distinguished, the algorithm starts with a very limited

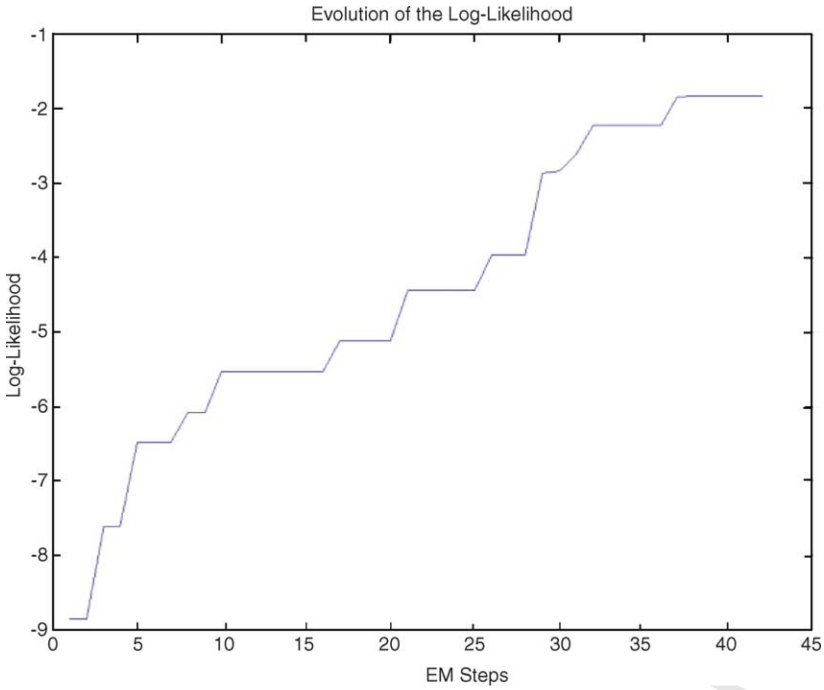

Fig. 1. Evolution of the log-likelihood for a situation with 10 different illumination conditions.

amount of clusters and calculates the log-likelihood for the current distribution model. New clusters are then included and the model is recalculated until the added log-likelihood for increasing the number of mixtures falls below a certain threshold. Fig. 1 shows the evolution of the log-likelihood for a situation where the algorithm correctly distinguished the 10 different illumination conditions which were applied to the object to be tracked. The result of this EM calculation is an $N_{\mathrm{LS}} \times N_{\mathrm{e}}$ light spectrum matrix L, with N_{LS} the number of illuminant spectra distinguished by the EM algorithm:

$$L = [\, q_{\mathrm{e}}^{\mathrm{T}}(1) \quad \cdots \quad q_{\mathrm{e}}^{\mathrm{T}}(n) \quad \cdots \quad q_{\mathrm{e}}^{\mathrm{T}}(N_{\mathrm{LS}})\,]^{\mathrm{T}}. \tag{8}$$

Together with the calculation of L, the nominal color for each of the clustered lighting conditions is calculated and stored in an $N_{\mathrm{LS}} \times 3$ color measurement matrix H_{N}. Fig. 2 shows the different nominal colors for an object under different illuminants. With the knowledge of H_{N} and L, we can calculate the inverse of the $N_{\mathrm{e}} \times 3$ reflectance spectrum matrix R:

$$R^{-1} \triangleq H_{\mathrm{N}}^{-1} \cdot L. \tag{9}$$

This R^{-1} matrix will be used to calculate the maximum a posteriori (MAP) distribution during the pixel classification process, as explained in the next paragraph.

2.2.2. Pixel classification

Now that we have estimates of the reflectance spectrum of the target object and now that we have obtained illuminant spectra corresponding to different lighting conditions, we want to correctly classify newly presented pixels as belonging to the target object or not, while keeping track of newly arising lighting conditions. The expectation Maximization algorithm provided us with 10 initial lighting conditions, which means that for every pixel, also 10 hypotheses for the lighting conditions will have to be calculated. We present a Bayesian solution

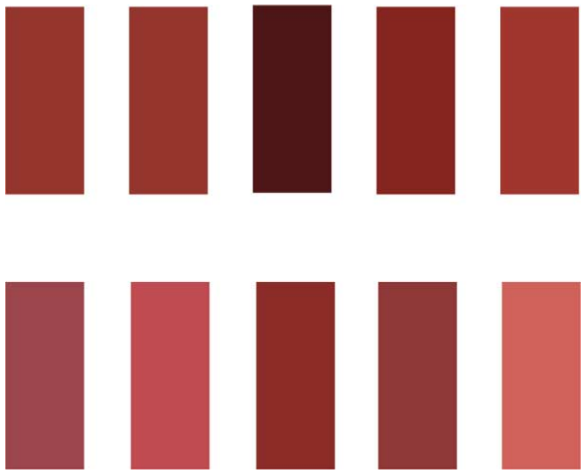

Fig. 2. Nominal colors for a red ball under different illumination conditions.

to solve these problems. New scene properties are brought into the model based upon the maximum a posteriori estimate of these parameters given the color measurements. When applying this classification, we search for the conditions that maximize $p(o = o_{\mathrm{Target}}, l, q_{\mathrm{e}}, \sigma|h)$ for any values of the lighting condition l, the illuminant spectrum q_{e} and the reflectance spectrum of the target object σ, given the color measurement triplet h:

$$[\hat{o}, \hat{l}, \hat{q}_{\mathrm{e}}] = \underset{[l, q_{\mathrm{e}}]}{\mathrm{argmax}}\ p(o, l, q_{\mathrm{e}}, \sigma|\hat{h}). \tag{10}$$

Using Bayes' rule, it can be shown that:

$$p(o, l, q_{\mathrm{e}}, \sigma|\hat{h}) \propto p(\hat{h}|q_{\mathrm{e}}, \sigma) \cdot p(q_{\mathrm{e}}|l) \cdot p(l) \cdot p(o). \tag{11}$$

We will now discuss the different factors in Eq. (11) and show how they can be calculated or estimated.

- $p(\hat{h}|q_{\mathrm{e}}, \sigma)$ is calculated by supposing that the measurements are corrupted by Gaussian noise:

$$p(\hat{h}|q_{\mathrm{e}}, \sigma) = \left(\frac{2\pi}{|\Sigma_{\mathrm{h}}|}\right)^{-3/2} \mathrm{e}^{-\|\hat{h}^{\mathrm{T}} - q_{\mathrm{e}}^{\mathrm{T}}, \sigma\|_{\Sigma_{\mathrm{h}}}}, \tag{12}$$

  where Σ_{h} is the measurement covariance matrix, $|\cdot|$ denotes the determinant and $\|\cdot\|\Sigma_{\mathrm{h}}$ is the Mahalanobis distance: $\|a\|_{\Sigma} = a^{\mathrm{T}}\Sigma^{-1}a$. The measurement covariance matrix is calculated together with the color measurement itself. To calculate the factor in the exponent, we record the nominal color values h_{N} of the perceived illuminants and these values are used to calculate the Mahalanobis distance to the current color triplet.
- $P(q_{\mathrm{e}}|l)$ represents the prior probability density of observing a certain illuminant spectrum q_{e}, given the lighting condition l. This is calculated during the Expectation Maximization phase of the learning process.
- $p(l)$ describes the prior probability of observing a certain illumination condition on a given point in the scene. There is no a priori knowledge about this, yet over time, it is possible to build up some knowledge about the different lighting situations at different points in the scene and this information can be used to derive a probability for the occurrence of lighting conditions in novel scenes. To do this, an illumination map of the surroundings of the target object is recorded. The values recorded in this map represent for each of the different possible illumination conditions, the probability that they would occur. These probabilities are calculated during the classification process using a voting system: a positive classification for a pixel given a lighting condition increases the probability for this lighting condition at this pixel position, while decreasing all other probabilities.

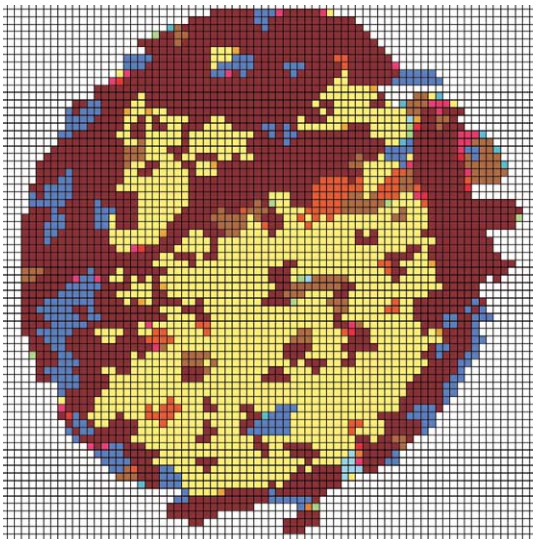

Fig. 3. Most probable lighting condition at each pixel (every color = different lighting condition).

The result of this process is illustrated in Fig. 3, which shows for each pixel which illumination condition is most likely to occur. As can be observed, there were two main illumination conditions present at this time instance: one near the central and lower right part and one near the top left part due to a shadowing effect. Near the edges, the influence of surface reflection causes other lighting conditions to occur.

- $p(o)$ represents the prior probability of observing the target object in the scene. This factor is estimated by dividing the number of pixels belonging to the target object, estimated at the previous time instance, by the total number of pixels in the image window. Fig. 4 shows how $p(o)$ stabilizes over time once the tracking is started.

Using these considerations, the pixel classification procedure calculates the probability for each pixel and labels the pixel as belonging to the target object or not based upon the result. Fig. 5 shows an example of a probability distribution for object presence calculated during the pixel classification process. The circular target object can clearly be identified when observing this distribution. Using this classification approach, the pixel classification is no longer performed directly based upon the pixels color value, as is classically done, but based upon the derived reflection characteristics, which makes the detection process very robust. This can also be observed by analyzing Fig. 6 which represents the unclassified pixels in gray and the classified pixels in black, both in the $l_1l_2l_3$ (left) and in the *RGB*-space (right). Fig. 6 shows that the applied classification strategy allows a large flexibility in the definition of the target objects color domain, as the classified pixels account for a considerable volume in both of the color spaces, while the false detection rate is kept low.

2.2.3. Model updating

During the actual tracking phase, the illumination model is continuously updated using Bayesian reasoning. The model updating stage estimates new lighting conditions together with their corresponding illuminant spectra. It is this procedure that ensures the adaptive nature of the pixel classification process within the general target-tracking program. The philosophy of this procedure is that we take a small patch from the target object (shown in Fig. 17 as the small square), try to recover the spectrum of the illuminant shining on this part of the target object and update our model if necessary. So, the first step in this process is to obtain a patch from the target object. For this, we cannot rely on the pixel classification process to tell us where the ball is, as in this case no new information would be added to

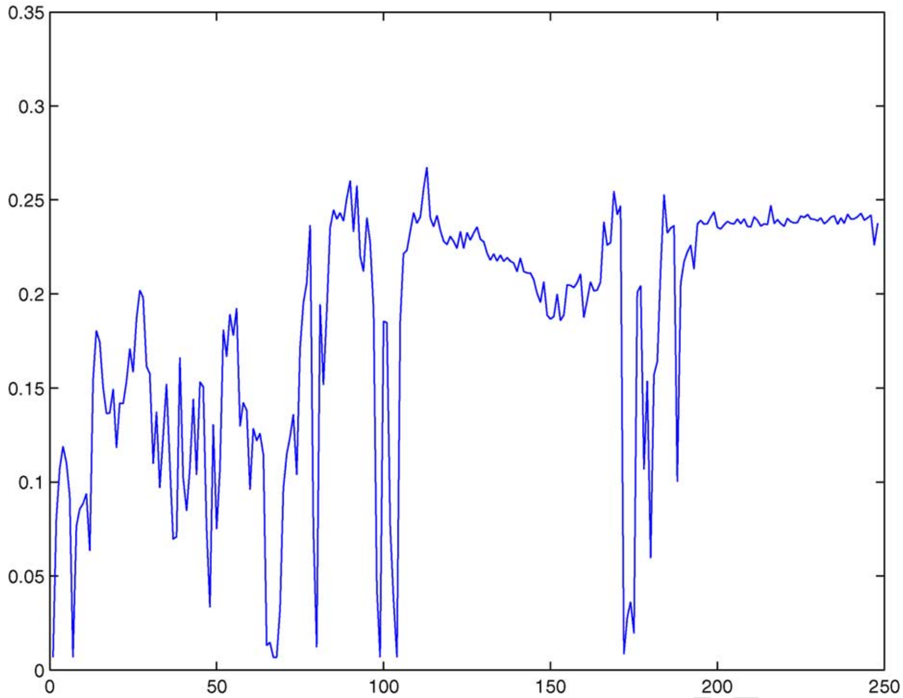

Fig. 4. Evolution of the probability of observing the target object in the scene.

the existing illumination model. The strategy here is to apply a circle or ellipse fitting upon the classified pixels and then to randomly select a patch within this circle or ellipse. For this patch, a nominal color h_N is calculated. If h_N is close to any of the mean h values of the already existing lighting conditions, no model updating is made. Otherwise, the new illumination condition is calculated and this new illumination condition will replace the one which was least used in the old model. After this, the probability of the new illumination condition is set to the mean of the others and the h_N values, covariance matrix and illumination maps are updated. This model updating algorithm does not need to run completely at every iteration, since there will no be no new illumination condition with every new

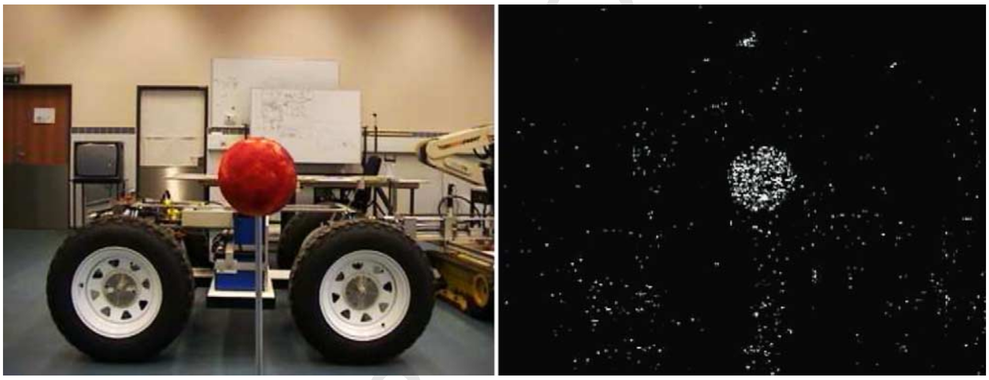

Fig. 5. One image frame and the corresponding probability distribution for object presence.

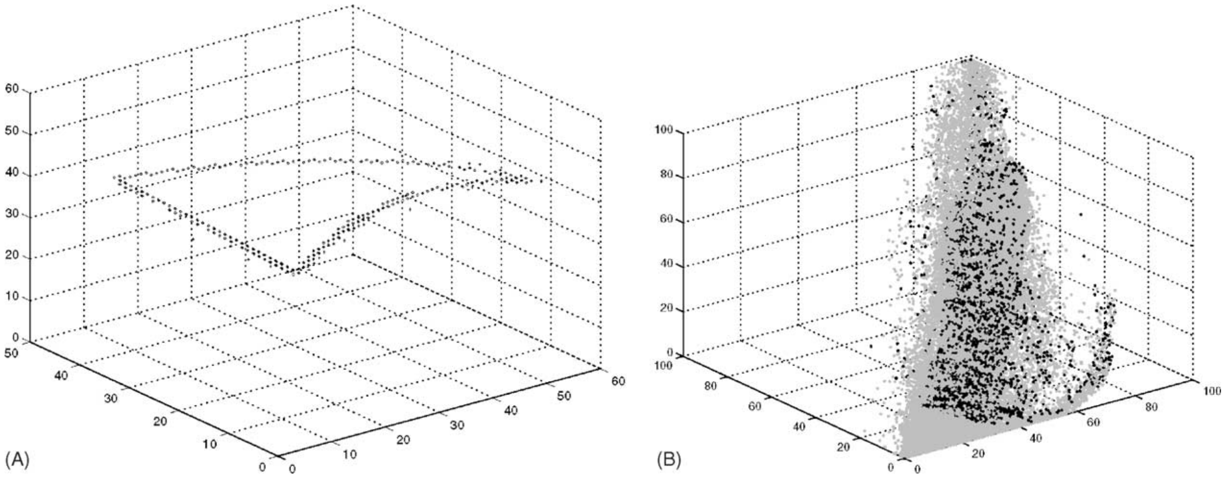

Fig. 6. Classification results of an entire image. Light gray dots represent unclassified pixels, whereas the black dots represent classified pixels in (A) $l_1l_2l_3$ and (B) *RGB*-space.

frame and only noteworthy changes in illumination will result in the model being updated. Therefore, the physical possibility of the proposed model update is tested considering the reflection characteristics of the target object, the change in illumination and the covariance on the measurements. The calculation of the new illumination condition itself can happen very rapidly, since we already know the reflectance spectrum matrix. After acquiring a nominal color triplet measurement h_N, we can write:

$$q_\mathrm{e}(N_\mathrm{new}) = h_\mathrm{N} \cdot R^{-1}, \tag{13}$$

N_new is the index of the rarest illumination condition within the L matrix, which will thus be replaced by the new lighting condition. R^{-1} is the pseudo-inverse of the reflectance spectrum matrix acquired during the learning phase. The performance of this model updating process is illustrated in Fig. 7. Fig. 7A shows the initial probability distribution for target object presence, while Fig. 7B shows the same distribution at a later time instance. This illustrates how the update step improves the Bayesian reflection model, such that the target object can be classified more clearly. To illustrate the adaptivity of the reflection model due to the updating step, Fig. 8 shows the pixel distributions at two different instances during a sequence, separated by a change in illumination conditions, as illustrated in Fig. 8A and B. In Fig. 8C and E, the initially classified pixels are represented in black and the

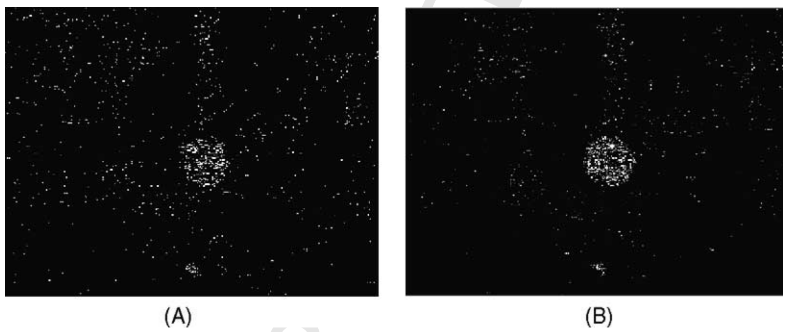

Fig. 7. Effects of model updating on the probability distribution for object presence.

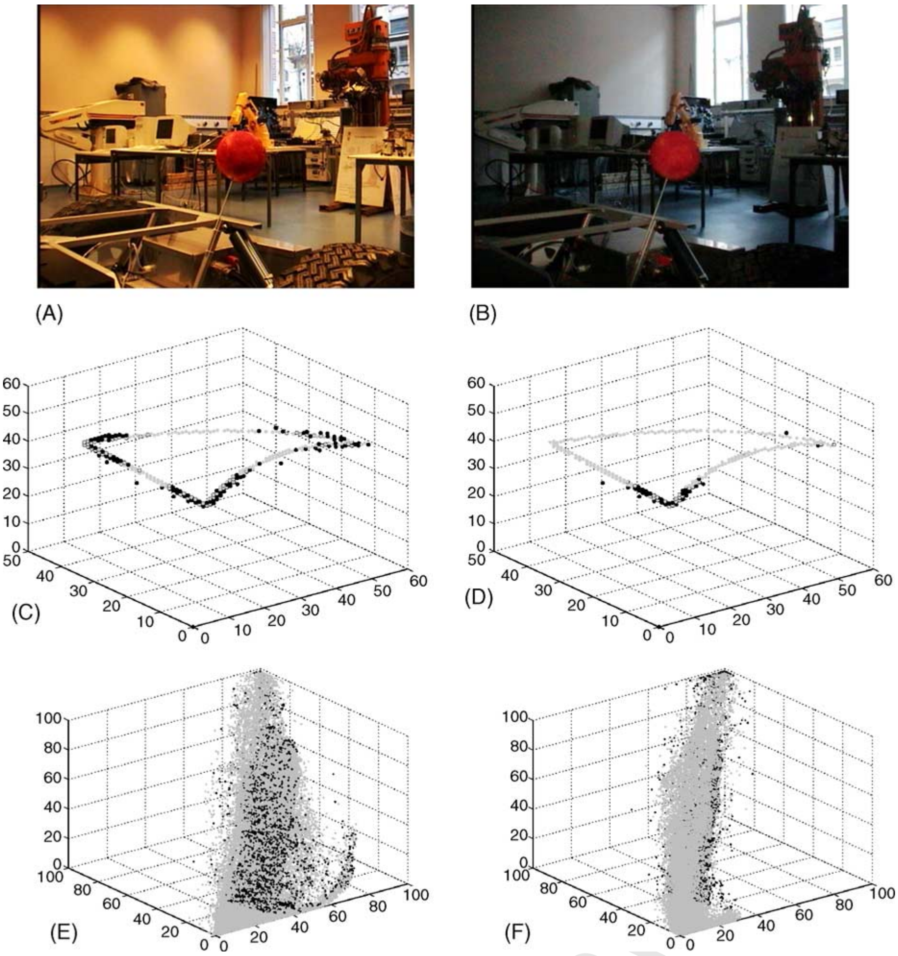

Fig. 8. Effects of illumination changes on the pixel distributions: (A) the original image with the target object (red ball) in front; (B) situation when the lights are turned off; (C) distribution of classified (black) and unclassified (light gray) pixels when the lights are on in $l_1l_2l_3$-space; (D) distribution of classified (black) and unclassified (light gray) pixels when the lights are off in $l_1l_2l_3$-space; (E) distribution of classified (black) and unclassified (light gray) pixels when the lights are on in *RGB*-space; (F) distribution of classified (black) and unclassified (light gray) pixels when the lights are off in *RGB*-space.

unclassified pixels in gray, respectively in the $l_1l_2l_3$ and the *RGB*-space, while Fig. 8D and F shows the same at a later time. As one can observe, the cluster of classified pixels has moved in the color space, together with the variation in illumination conditions. These figures show also very clearly the advantage of working with the $l_1l_2l_3$ color space instead of the *RGB*-space, while the general distribution of pixels for this first one stays more or less the same under illumination shifts, whereas the *RGB*-space suffers from dramatic changes. Another fact is that it is not straightforward to accord a color cluster in the *RGB*-space to a certain reflective surface, whereas this is far easier in the $l_1l_2l_3$ color space.

The preceding discussion shows how we can acquire a description for the color of an object which is quite independent of the illumination conditions. Now, the object can be identified reliably and tracked in a following stage, as we will explain in the next section.

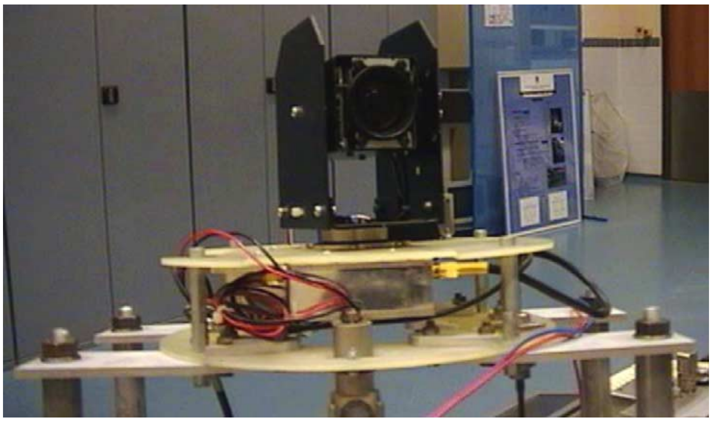

Fig. 9. The pan–tilt camera system used for visual servoing.

3. Camera control for target tracking

3.1. System overview and setup

The application for this work concerns the use of a pan–tilt camera to track and to estimate the position of a target object. This problem is solved as a visual servoing problem, combining image processing, kinematics, dynamics, control theory and real-time computing. The camera system used for this purpose is shown in Fig. 9. The camera platform consists of two servomotors. One is under the camera and controls the pan angle. The other one is on the camera side and controls the tilt angle.

To define the different system parameters present in the visual feedback loop, the camera control parameters must be defined first. We use the pinhole camera model and map the 3D world coordinates onto the image plane using the perspective projection. Now, let us consider a point *P* in the world coordinate system and its projection in the image *p*, as shown in Fig. 10. The point *p* is given by $(u, v) = (|ox_1|, |oy_1|)$. The reciprocal values of pixel size (d_x, d_y), the camera focal length *f* and the principal point $o(o'_u, o'_v)$ are known from the camera calibration step.

In Fig. 10 we define two angles:

$$\alpha = \angle ocx_1, \tag{14}$$

$$\beta = \angle ocy_1. \tag{15}$$

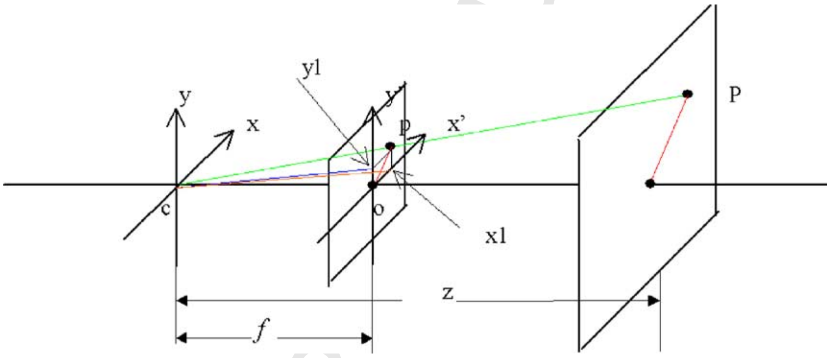

Fig. 10. Definition of the camera control parameters α and β.

These two angles represent the difference in orientation between the optical axis and the line *cpP*. We can calculate α and β by

$$\alpha = \tan^{-1}\left(\frac{u - o'_u}{f \cdot d_x}\right), \tag{16}$$

$$\beta = \tan^{-1}\left(\frac{o'_v - v}{f \cdot d_y}\right). \tag{17}$$

Our aim is to keep the target center coincident with the image center, thus α and β will define the pan and tilt control parameters of the camera.

We define the servomotor–target–camera system as our plant. The above defined angles are used for camera control and subsequently for target tracking. The plant is considered as a time-variant system due to the unknown motion of the target. The target movement is estimated in real-time and considered in our system as the plant state transition of free response. Note that Eqs. (14)–(17) underline the non-linear character of the proposed plant model.

In order to meet the system dynamic characteristic requirements, a two-phase control strategy was implemented with a separate initialization phase and an observer-based full-state feedback control phase. During the system initialization phase a Proportional and Integral regulator (PI regulator) is used to track the target. At the same time, the plant input and output data are collected to identify the plant model and to train the state observer and all the adaptive filters used in the system. The plant model will be used in state observation and state feedback control. After a certain period of time, the system control strategy is switched from phase one into phase two: the full-state feedback control state.

3.2. Target tracking during initialization

During initialization, the system (camera) is controlled by a PI regulator designed for target tracking. The system is considered as a time invariant one and the target movement is considered as an environment disturbance to the system. The block diagram of the control system for this phase is given in Fig. 11. An error signal e composed by comparing the image center o and the camera's output y, i.e. the previous target image center. Based upon this error signal, the PI regulator calculates a new control signal u fed to the camera servo control system, which results in a movement of the camera optical axis m. The target movement v will induce noise, which is represented in Fig. 11 as n. $F(v)$ is the transfer function representing the relationship between v and n. The superposition of the noise signal n and the movement of the optical axis of the camera m, provides the input for the optical system of the camera, which will calculate a new target image center y. Because the servomotor system of the camera is a closed-loop control system and can roughly be considered as a second-order system, it can be controlled by a PI regulator by finding the system poles. Using this control method, the camera can start tracking right away, while the plant model is being built up from zero, as we explain in the following section.

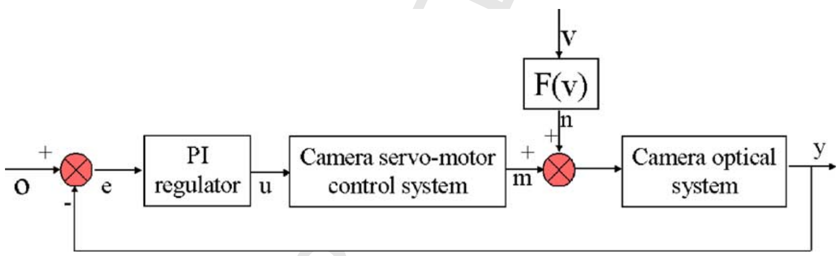

Fig. 11. Initialization system block diagram.

3.3. Plant model identification

The dynamic properties of our system can be described by the following set of non-linear differential equations [25]:

$$\dot{x}(t) = f(x(t), u(t), t), \tag{18}$$

where $x(t) \in \Re^n$ is the state vector, $u(t) \in \Re^m$ is the input vector and f is a mapping $\Re^n \times \Re^m \to \Re^n$ defined as

$$f(x(t), u(t), t) = \begin{bmatrix} f_1(x(t), u(t), t) \\ f_2(x(t), u(t), t) \\ \vdots \\ f_n(x(t), u(t), t) \end{bmatrix}. \tag{19}$$

The existence and uniqueness of the solutions are assumed. This means that for a given system state $x(t)$, there exists a unique input $u(t)$. For our system, these assumptions are only guaranteed within the operational limits of the pan–tilt unit and assuming that, for a short period, the plant is time invariant. This last requirement is fulfilled when the speed of the control system is much quicker than the speed of the plant parameter's changing. To establish a practically useful plant model we must apply a linearization around the equilibrium point (x_0, u_0) where both x_0 and u_0 are zero. In our control strategy for target tracking, we try to keep the target center and the image center coincident, so we can always linearize the non-linear dynamic system around the equilibrium point. Moreover, when we apply the system identification, under the condition of weak perspective (small view-angle) all the requirements of linearization are met. Therefore, we can use a linear model to approximate our plant dynamics. For a discrete time system, the corresponding function can be written as

$$x(k+1) \approx A \cdot x(k) + B \cdot u(k). \tag{20}$$

The matrices A and B are time-dependent, so the corresponding linear systems is a time-variant one.

The system model represented in Fig. 12 is mathematically expressed as

$$X(k+1) = A(k) \cdot X(k) + B(k) \cdot u(k) + W(k), \tag{21}$$

$$y(k) = C(k) \cdot X(k) + v(k). \tag{22}$$

In Fig. 12 and Eqs. (21) and (22), $X(k)$ represents the system state vector consisting of the angular position and angular velocity of the target, $y(k)$ the system output representing the difference between the camera principal point

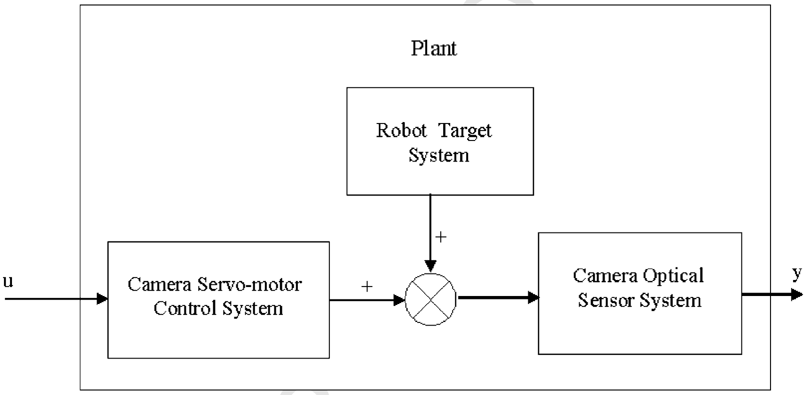

Fig. 12. Dynamic system model.

and the target image position, $A(k)$ the plant system matrix, $B(k)$ the plant input matrix, $C(k)$ the plant output matrix, $W(k)$ the model noise vector, whereas $v(k)$ the measurement noise variable and $u(k)$ the system control input.

To estimate the system model in real-time, we simplified the plant model by using a second-order difference model (the projection on a subspace) to approximate the real system model (a multifold space curve) at each sampling point. This reduces the model error significantly. Higher-order system models introduce noise into the control system and make it more difficult to control. For our application, we also assume that the movement of the target does not change abruptly (the motion acceleration is considered small). Therefore, we can just select the angular position and the angular speed of the target as state variables (the eigenvectors which correspond to the most significant eigenvalues in the discrete system state space). From the point of view of pole position in the s-plane, this is equivalent to keeping the plant's main poles and omitting its other poles. The other poles are often far away from the imaginary axis and their influence in the output will die out very quickly. The parameters of the plant state space function and the plant output function can then be written as

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_0 & -a_1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k), \tag{23}$$

$$y(k) = \begin{bmatrix} c_0 & c_1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}, \tag{24}$$

(x_1, x_2) is the state vector corresponding to one of the camera angles (pan or tilt) and the corresponding angular velocity. (a_0, a_1, c_0, c_1) are the system parameters to be estimated.

We use a least-mean-square (LMS) second-order adaptive filter as plant parameter estimator [14]. The same structure for the LMS filter is used for both pan and tilt plant parameter estimation. The estimator works in two steps. First, it uses the updated input data, output data and filter's tap weights to estimate the system current output value. In the second step, it uses the updated input data, output data and the error between the estimated current output and the real output of the system to modify the tap weights $w(k)$ of the filter. These updated tap weights are our plant parameter's estimates. As an example, the LMS adaptive filter for the plant parameter's estimation of the pan is presented here. For this, the estimation error is defined as

$$e(k) = d(k) - y(k). \tag{25}$$

With $d(k)$ the desired output at instant k, being the real target position in the X(pan)-direction at instant k. $y(k)$ is the estimated output at instant k. The cost function is defined as

$$J(k) = \tfrac{1}{2} E[|e(k)|^2]. \tag{26}$$

The purpose of the filter is to minimize $J(k) \to J_{\min}$. A second-order filter is used. The tap weight vector of the filter is defined as

$$w(k) = [-\hat{a}_1(k) - \hat{a}_o(k)\hat{c}_1(k)\hat{c}_0(k)]^{\mathrm{T}}. \tag{27}$$

The filter's input vector is made up of the past plant output and the past plant control command:

$$u(k) = [d(k-1)d(k-2)u(k-1)u(k-2)]^{\mathrm{T}}, \tag{28}$$

where $u(k)$ is the control signal for the X direction at instant k.

The filter can now be defined by the following set of iteration functions:

$$y(k+1) = \hat{w}^{\mathrm{T}}(k) \cdot u(k), \tag{29}$$

$$e(k) = d(k) - y(k), \tag{30}$$

$$\hat{w}(k+1) = \hat{w}(k) + \mu(k) \cdot u(k) \cdot e(k), \tag{31}$$

where $\mu(k)$ is the step-size parameter. Having estimated the plant parameters, one can estimate the matrices of the plant state space model from instance k to instance $k+1$:

$$A(k+1,k)=\begin{bmatrix}0 & 1\\ -\hat{c}_0(k) & -\hat{c}_1(k)\end{bmatrix}, \tag{32}$$

$$B(k+1,k)=\begin{bmatrix}0\\ 1\end{bmatrix}, \tag{33}$$

$$C(k+1,k)=\begin{bmatrix}\hat{b}_0(k) & \hat{b}_1(k)\end{bmatrix}. \tag{34}$$

It should be noted that the LMS adaptive filter can only be used for non-stationary systems. Therefore, we suppose that the target movement can be modeled as a non-stationary Markov process.

3.4. Full-state feedback control

The second phase control strategy consists of an observer-based full-state feedback control strategy. We use an on-line identification method to identify in real-time the plant model and apply the identified model in the Kalman observer to emphasize the influence of the change of plant model on the plant state estimation. At the same time, the estimated state models are used for the state feedback strategy calculation to emphasize the time-variant property of the control system. The main tasks of this phase are observing the plant states, calculating the feedback control value and identifying the plant model, as shown in Fig. 13.

Now that the plant model has been identified, its state vector will be estimated using Kalman filtering [14]. The Kalman filter works as a current observer, as shown in Fig. 14. It takes into account the dynamics of the target's movement by using the time-variant plant model. The reason for using a Kalman filter as an observer is mainly to reduce the influence of noise that comes from both the measurement inaccuracy and the model inaccuracy. From Fig. 14, we can see that the state observer is a dynamic system. It takes the plant input and output as its input and the estimated plant states as its output. In Fig. 14, u represents the plant input signal (the camera pan or tilt control signal), y is the plant output signal (the angle estimated from the image), $\tilde{x}$ is the estimated plant state vector, $A(k+1,k)$ is the plant system transition matrix from instant k to instant $k+1$, $B(k+1,k)$ is the plant control input matrix from instant k to instant $k+1$, $C(k+1,k)$ is the plant output matrix from instant k to instant $k+1$. The plant model can then be written as

$$x(k+1)=A(k+1,k)\cdot x(k)+B(k+1,k)\cdot u(k)+v_1(k), \tag{35}$$

$$y(k)=C(k+1,k)\cdot x(k)+v_2(k). \tag{36}$$

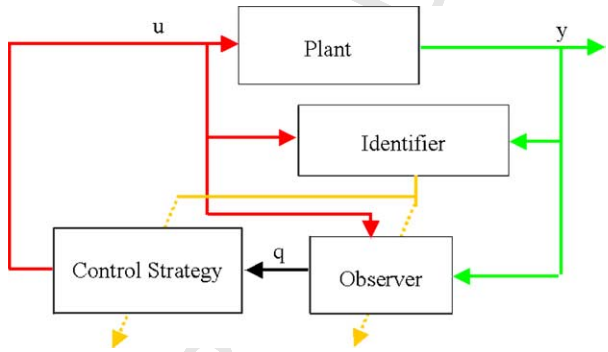

Fig. 13. Observer-based full-state feedback control.

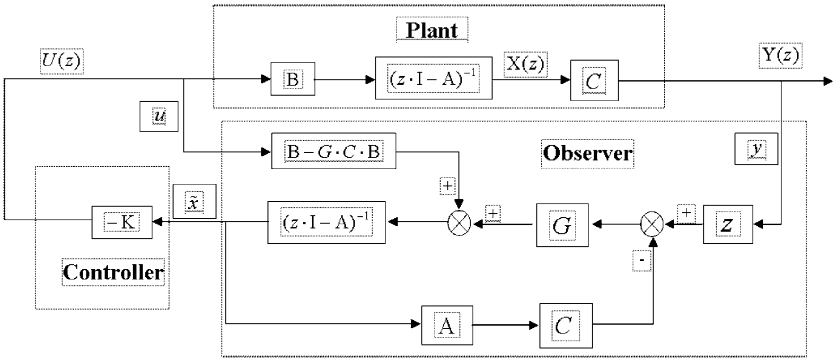

Fig. 14. The observer-based full-state feedback control system.

In Eqs. (35) and (36), $v_1(k)$ and $v_2(k)$ represent respectively the system process noise and the observation noise added to the plant model. We chose the pole assignment method to design the state feedback controller. The pole assignment method is a method in which the closed-loop system poles of a time-variant system can be kept in the desired constant positions with system state feedback. For a control system, the knowledge of the closed-loop system poles' positions induces the knowledge of the characteristics of the system.

First, we can set the poles' positions in the primary strip (from the sampling frequency) of the s-plane, according to the needed system dynamic characteristics (the response frequency and decay speed). These poles can be used as a design guideline. With the values of the two poles (s_1, s_2) and with the knowledge of the sampling period T_s, we can estimate the position of the poles (z_1, z_2) of the corresponding linear discrete time invariant system in the z-plane. This information will be used in the estimation of the feedback gain of the feedback controller. For this purpose, we go out from the equation giving the control input in a full-state feedback control scheme, given by

$$u(k) = -K \cdot x(k). \tag{37}$$

This function is integrated in the state space function of the plant, given by Eq. (20), such that we get the closed-loop state function of the full-state feedback control system:

$$\dot{x}(k+1) = (A - BK) \cdot x(k). \tag{38}$$

From Eq. (38), we can see that the closed-loop system characteristic function is

$$\psi_{\text{sys}}(z) = |z \cdot I - A + B \cdot K| = (z - \lambda_1) \cdot (z - \lambda_2) \cdot (z - \lambda_n), \tag{39}$$

where $\lambda_{i=1,\dots,n}$ are the poles of the closed-loop system.

According to the system dynamic characteristics we need, we can specify the desired poles' positions on the right-hand side of Eq. (39) and solve Eq. (39) for the given control strategy K. Thus, we use the estimated control strategy K to perform the full-state feedback control of the system given by Eq. (38). In our application this is realized in the following way. At each step i we specify a feedback gain matrix for the second-order system:

$$K(i) = [K_1(i) K_2(i)]. \tag{40}$$

This feedback gain matrix determines how to use every state of the plant in the control signal to keep the poles' positions of the closed-loop system time invariant:

$$u(i) = -\begin{bmatrix} K_1(i) & K_2(i) \end{bmatrix} \begin{bmatrix} x_1(i) \\ x_2(i) \end{bmatrix}. \tag{41}$$

The second term in Eq. (38) can now be written as

$$B(i+1\cdot i)\cdot K(i)\begin{bmatrix} x_1(i)\\ x_2(i)\end{bmatrix}=\begin{bmatrix} 0 & 0\\ K_1(i) & K_2(i)\end{bmatrix}\begin{bmatrix} x_1(i)\\ x_2(i)\end{bmatrix}. \tag{42}$$

We now substitute Eq. (42) into Eq. (38) and use the result of the system identification (Eq. (32)) to write the transition matrix A. The closed-loop system matrix now becomes

$$A(i+1,i)-B(i+1,i)\cdot K(i)=\begin{bmatrix} 0 & 1\\ -(\hat{a}_0(i)+K_1(i)) & -(\hat{a}_1(i)+K_2(i))\end{bmatrix}. \tag{43}$$

The system characteristic function of the closed-loop system is then:

$$\psi_f(z)=|zI-A(i+1,i)+B(i+1,i)\cdot K(i)|=z^2+(\hat{a}_1(i)+K_2(i))\cdot z+(\hat{a}_0(i)+K_1(i))=0. \tag{44}$$

By considering $\psi_{\mathrm{req}}(z)=\psi_f(z)$, the required gain is obtained:

$$K_{j+1}(i)=\alpha_j-\hat{a}_j(i),\quad j=0,1. \tag{45}$$

The plant characteristic function's parameters of the ith step have been estimated during the initialization step, α_1 and α_0 have been estimated from the pole assignment step, thus Eq. (45) can be used to solve the needed feedback gain.

3.5. Windowed tracking

In order to increase the tracking sampling rate and the signal-to-noise ratio of the camera control, a bounding box (search window/region of interest) around the target image is defined. An LMS filter is used to estimate and to predict the position $(\bar{x}, y)$ and size (l, h) of the defined search window, taking into account the activity of the camera. The window size is calculated by using the second-order moments of the detected target boundary (μ_x^2, μ_y^2):

$$l=C_1\cdot\mu_x^2+2\cdot\varepsilon, \tag{46}$$

$$h=C_2\cdot\mu_y^2+2\cdot\varepsilon, \tag{47}$$

where C_1 and C_2 are scale factors and ε is tolerance.

The prediction of the search window position and size are made during the tracking process. Therefore, the time-variant characteristics of the system and the camera activity are taken into account. The structure of the adaptive LMS filter used for the purpose of predicting the search window position is identical to the one for predicting the search window size. The desired system outputs $d(k)$ are defined as the real search window position $(\bar{x}, y)$ and size (l, h). The predictor works in two steps. First, it uses the old input data and the current desired output data to train the filter; that is, to update the filter tap weights. In the second step, it uses the updated input data and tap weights to estimate a prediction for the real coming output. Note that the working principle is different from the LSM filter used for the system identification, although the prediction error and the cost function are defined similarly according to Eqs. (25) and (26). Supposing that the filter is of Mth-order, we define the tap weight of the filter as

$$w(k)=\left[\hat{w}_0(k)\hat{w}_1(k)\quad \hat{w}_{M-1}(k)\right]^{\mathrm{T}}. \tag{48}$$

The input vector is

$$u(k)=\left[u(k)u(k-1)\quad u(k-M+1)\right]^{\mathrm{T}}. \tag{49}$$

For the estimation of the new search window position, $u(k)$ is the difference between $\bar{x}(k)$ or $\bar{y}(k)$ and the control command: $u(k)=\bar{x}(k)-x_{\mathrm{co}}(k)$ or $u(k)=\bar{y}(k)-y_{\mathrm{co}}(k)$, where $x_{\mathrm{co}}(k)$ and $y_{\mathrm{co}}(k)$ are the camera control signals

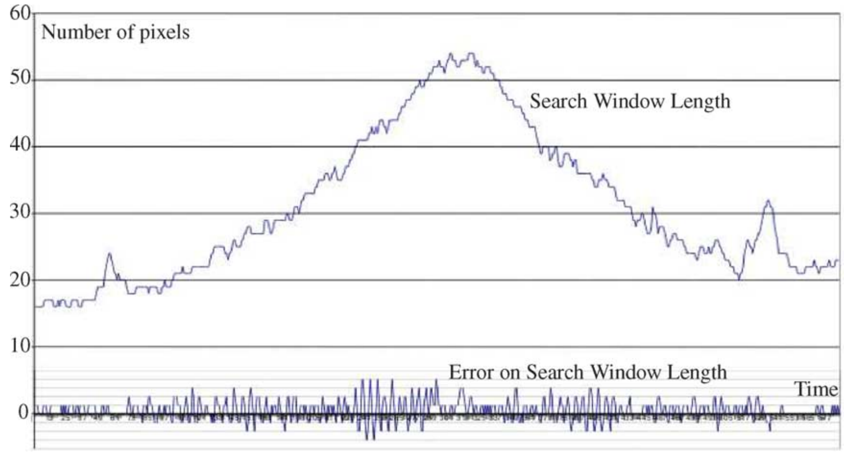

Fig. 15. Search window size prediction and associated error for a horizontal pass-by test.

respectively in the X and in the Y direction. For the new search window size, $u(k) = \mu_x^2(k)$ or $u(k) = \mu_y^2(k)$. Therefore, difference between the filters used for search window position and size estimation lies in the fact that the first one uses the window position and the camera control signal as inputs to return a new window position estimate, whereas the second one uses the second-order moments as inputs to calculate the window size. Experimentally, a second-order filter was chosen, because it proved to allow a stable and fast tracking behavior.

The search window prediction results can be analyzed in Fig. 15, which shows the prediction of the search window size and the associated error. During this test, the target object was mounted on a robot arm and it first moved towards the camera and then away from it. This horizontal movement caused especially the window size to change: as we can see the search window becomes larger when the target is closer to the camera and smaller when the target moves away. The noise pulses are caused by the background of the test scene. The prediction error is always small compared to the actual value of the window size.

3.6. Target position estimation

Target location estimation is an extremely important subject in robotic applications. The visual servoing system presented here involves a method for estimating the target position, i.e. the quantitative description of where the target is with respect to the observers view. For our application, the similarity of the target shape and its projected image is used to estimate the camera–target distance. The origin of world frame is set at the center of the camera. The camera platform is kept horizontal. Then, the position of the target can be described by three parameters: the horizontal angle, the vertical angle and the distance between camera and target. Angles are calculated using the pose of the camera and the orientation angles of the target image in the camera coordinate system. The distance between camera and target is estimated by comparing the size of the target shape in the image window to the known dimensions of the target object, taking into account the effective camera focal length. We incorporated several improvements for the important distance estimation step, as this is an operation which is highly sensitive to several kinds of noise. One improvement is to make use of a low-pass band filter. However, the largest increase in precision could be achieved by considering only circular objects and by introducing circle and ellipse fitting procedures to more accurately measure the radius of the circular target object in the image plane. For ellipse fitting, a very fast algorithm, described in [32], was used. The circle fitting procedure is slightly more precise, but is much slower, since it relies on a heuristic brute force approach to find the best fit. Fig. 16 compares the capabilities of the circle and ellipse fitting procedures in normal and in noisy conditions. It clearly shows that the circle fitting procedure is

Fig. 16. Comparison between the circle and ellipse fitting procedures. Whitened pixels mark positive classifications. Ellipses are marked with a blue line, while circles are filled in green. (Top left) ellipse fitting in normal conditions; (top right) circle fitting in normal conditions; (bottom left) ellipse fitting in noisy conditions; (bottom right) circle fitting in noisy conditions.

capable of producing better matches for the object to be tracked, yet as this process requires also more calculation time, its use is limited by the available computing resources.

4. Experimental results

We have previously shown in Fig. 5 the result of the pixel classification procedure. As can be seen, the target object (a ball) is very clearly visible and the falsely classified pixels can easily be filtered out by subsequent erosion and dilation operations on the created binary image.

Comparing the used approach to other scientific work is difficult, because on the subject of tracking the presented classification algorithm does not take into account any other parameters (e.g. shape or texture) than the color attributes like other authors have done. On the subject of color constancy, the presented algorithm is not able to deliver the high-quality data about the illuminant spectrum like other, more time consuming methods, are capable of. Fig. 17 shows the strength of the presented color constancy algorithm by comparing it to another real-time color-constancy approach. The middle row shows two pictures shot during the same sequence, but with a difference in illumination conditions (lights turned off). On the top row, you can see the results the gray world algorithm returns for these images. This simple algorithm goes out from the assumption that the average of all colors in an image is gray, so the red, green and blue components of the average color are equal. The amount the image average departs from gray determines the illuminant *RGB*. On the bottom row, you can observe the classification results of the presented color constancy technique. As you can observe by noticing the whitened pixels which

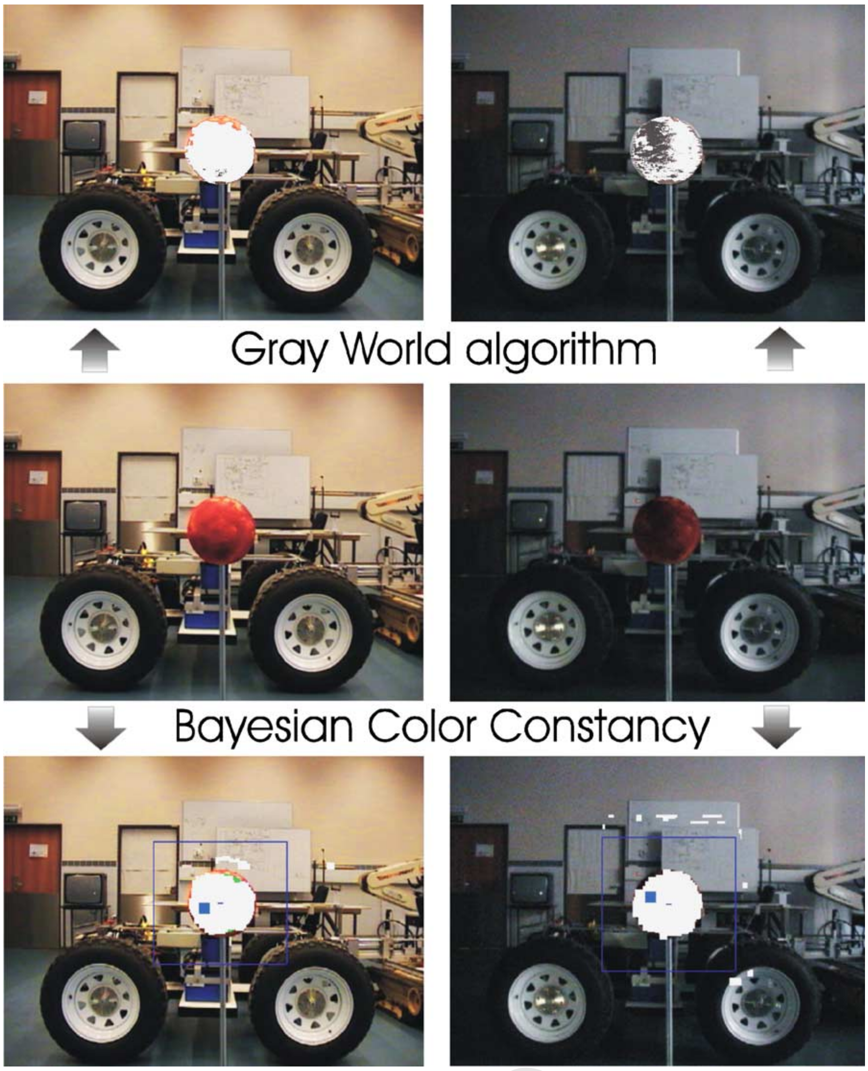

Fig. 17. Comparison of color constancy approaches. (Middle row) two pictures shot during the same sequence (lights on/off); (top row) classification result of the Gray world algorithm; (bottom row) classification result of the presented color constancy technique.

indicate that a target has been found here, the algorithm succeeds in recognizing and classifying the searched object.

Fig. 18 shows the tracking error in the *X* direction and demonstrates the tracking ability of this system. This data was recorded during the same test already explained in the section about windowed tracking (target first moving towards the camera, then away from it). Notice how the error increases when the target moves closer to the camera; it decreases when the target moves away from the camera. This behavior is caused by the inertia of the tracking system (the pan–tilt camera). Fig. 19 gives an example of the variation of the absolute distance errors over a number of samples for a target located at a distance of about 7 m. Concerning the real-time capabilities, the target-tracking program is able to run at about 10 fps on a PC equipped with an 1.7 GHz PIV processor, which is adequate for most everyday target-tracking tasks.

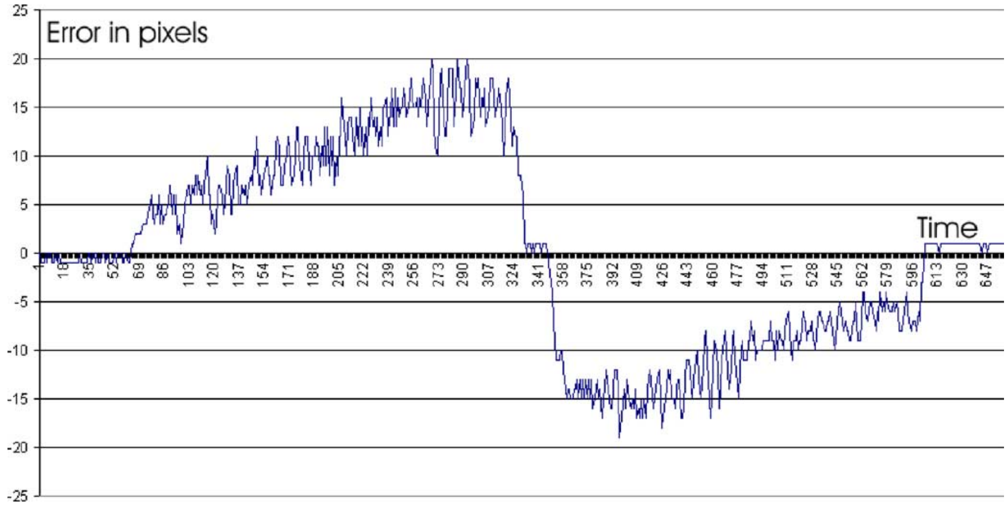

Fig. 18. Tracking error in pixels during a horizontal pass-by test.

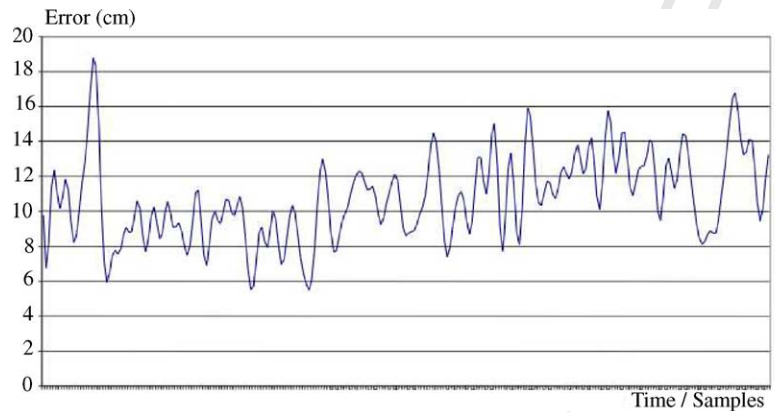

Fig. 19. Variation of absolute distance error in cm over time for a target at about 7 m.

5. Conclusions

We have shown a powerful set of algorithms, which were combined to form a universally useable system for automated target detection, tracking and position estimation, using a single and fairly simple pan–tilt camera. The main importance of this work is that we have shown that it’s feasible to integrate the benefits of different techniques, while avoiding their drawbacks.

The Bayesian-based color constancy approach which was used, ensures that this system can keep working, even in harsh illumination conditions. Color constancy has so far been a field mainly focussed at processing static images, yet also due to the increasing computing power, it now becomes an option for real-time applications too. Here, we have shown an algorithm which uses Bayesian reasoning to cope with changing illumination conditions. The presented technique is not able to produce quality data about the illuminant spectrum, it just aims to retrieve a reliable description of the reflection characteristics of the object to be tracked. This compromise we made here does not have any negative affect for the visual servoing program as a whole, as the knowledge of the illuminant spectrum is not really necessary for this.

In the field of camera control, we have tried to integrate the benefits of image-based and position-based visual servoing approaches. The tracking algorithm takes advantage of the speed of the image-based approach because it calculates the control signals based upon features in the 2D image space. On the other hand, the 3D target position is calculated in a separate procedure, enabling the output of high-quality 3D-positioning data, as in the position-based visual servoing approach. A main disadvantage of this latter technique was also the need for a precise model with a lot of a priori knowledge, whereas the image-based approaches could do without a model. In this work, a two-phase approach was chosen, where in the beginning a model-free tracking technique is used and later a model-based technique. This setup allows the servoing system to work under all circumstances without the need for any prior knowledge, as the system model can be built up during the initialization phase. The system identifier, Kalman filter-based system state observer and the controller itself have been explained in the article and the way they control the pose of the camera coordinate system to track the target. The online identification method is used to deal with time-variant problems. The poles' position method is used to guarantee the system's stability and the quality of the system's response. As the target is a time-variant system, both the identifier and the observer need some time to follow the system changes. Therefore, the tracking results have some biases, however this bias is reduced by using the window tracking method.

This research was specifically aimed at applicability in the field of robotics, yet due to its general structure it can be used for a wide range of applications.

Acknowledgements

This research has been conducted within the framework of the Inter-Universitary Attraction-Poles program number IAP 5/06 Advanced Mechatronic Systems, funded by the Belgian Federal Office for Scientific, Technical and Cultural Affairs.

References

[1] K. Barnard, Practical colour constancy, Ph.D. Thesis, Simon Frasier University, School of Computing, 1999.

[2] M.H. Brill, G. West, Chromatic adaptation and color constancy: a possible dichotomy, Color Research Applications 11 (1986) 196–204.

[3] E. Cervera, F. Berry, P. Martinet, Image-based stereo visual servoing: 2D vs 3D features, in: Proceedings of the 15th IFAC World Congress on Automatic Control, IFAC'02 Barcelona, Spain, July 2002.

[4] Y.T. Chan, A.G.C. Hu, J.B. Plant, A Kalman filter based tracking scheme with input estimation, IEEE Transactions on Aerospace and Electronic Systems 15 (2) (1979) 237–244.

[5] CIE, Colorimetry, Bureau Central de la CIE, 1986.

[6] J. Cohen, Dependency of the spectral reflectance curves of the Munsell color chips, Psychon. Sci. 1 (1964) 369–370.

[7] P.I. Corke, Visual control of robot manipulators—a review, in: K. Hashimoto (Ed.), Visual Servoing, 1994.

[8] F.X. Espiau, E. Malis, P. Rives, Robust features tracking for robotic applications: towards 2 1/2 D visual servoing with natural images, in: Proceedings of the IEEE International Conference on Robotics and Automation, vol. I, Washington, USA, May 2002, pp. 574–579.

[9] B.V. Funt, V. Cardei, K. Barnard, Learning color constancy, in: Proceedings of the IS and T/SID Fourth Color Imaging Conference: Color Science, Systems and Applications, 1996, pp. 58–60.

[10] B.V. Funt, M.S. Drew, J. Ho, Color constancy from mutual reflection, International Journal of Computer Vision 6 (1) (1991) 5–24.

[11] J. Gangloff, M. de Mathelin, G. Abba, 6 d.o.f. high speed dynamic visual servoing using GPC controllers, in: Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 1998, pp. 2008–2013.

[12] T. Gevers, A.W.M. Smeulders, Color-based object recognition, Pattern Recognition 32 (1999) 453–464.

[13] T. Gevers, H. Stokman, Reflectance Based Edge Classification, Vision Interface, Trois-Rivires, Canada, May 1999.

[14] S. Haykin, Adaptive Filter Technology, Prentice-Hall, 1996. ISBN 0130901261.
[15] J. Hill, W.T. Park, Real time control of a robot with a mobile camera, in: Proceedings of the Ninth International Symposium on Industrial Robots, March 1979, pp. 233–246.
[16] J. Ho, B.V. Funt, M.S. Drew, Separating a color signal into illumination and surface spectral components, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (10) (1990) 966–977.
[17] D.B. Judd, D.L. MacAdam, G.W. Wyszecki, Spectral distribution of typical daylight as a function of correlated color temperature, Journal Optical Society of America 54 (1964) 1031–1041.
[18] E.H. Land, J.J. McCaan, Lightness and the retinex theory, Journal Optical Society of America 61 (1971) 1–11.
[19] M. Loser, N. Navab, B. Bascle, R. Taylor, Visual servoing, a new image-guided technique for automatic and uncalibrated percutaneous procedures, in: Proceedings of the Medical Imaging 2000, San Diego, CA, USA, February 2000.
[20] R.C. Love, Surface reflection model estimation from naturally illuminated image sequences, Ph.D. Thesis, University of Leeds, September 1997.
[21] E. Malis, Survey of vision-based robot control, Technical Report, INRIA, Sophia Antipolis, France.
[22] L.T. Maloney, Evaluation of linear models of surface spectral reflectance with small numbers of parameters, Journal of the Optical Society of America A3 (1986) 1673–1683.
[23] L.T. Maloney, B. Wandell, Color constancy: a method for recovering surface spectral reflectance, Journal of the Optical Society of America A 3 (1) (1986) 29–33.
[24] J. Matas, R. Marik, J. Kittler, Illumination invariant colour recognition, in: Proceedings of the Fifth British Machine Vision Conference, 1994.
[25] T.P. McGarty, Stochastical Systems and State Estimation, Wiley, 1974. ISBN 0-471-58400-2.
[26] T. Mitsuda, Y. Miyazaki, N. Maru, F. Miyazaki, Precise planar positioning method using visual servoing based on coarse optical flow, Journal of the Robotics Society of Japan 17 (2) (1999) 71–77.
[27] J. Piepmeier, A dynamic quasi-Newton method for model independent visual servoing, Ph.D. Thesis, George W. Woodruff School of Mechanical Engineering, June 1999.
[28] A. Ruf, R. Horaud, Vision-based guidance and control of robots in projective space, in: Proceedings of the Sixth European Conference on Computer Vision (ECCV'00), vol. II of Lecture Notes in Computer Science, Dublin, Ireland, June 2000, pp. 50–83.
[29] A.C. Sanderson, L.E. Weiss, Image-based visual servo control using relational graph error signals, in: Proceedings of the IEEE, 1980, pp. 1074–1077.
[30] S.A. Shafer, Using color to separate reflection components, in: Color Research and Application, vol. 10, no. 4, 1985, pp. 210–218 (also available as Technical Report TR-136, Computer Sciences Department, University of Rochester, NY, April 1994).
[31] Y. Tsin, R. Collins, V. Ramesh, T. Kanade, Bayesian color constancy for outdoor object recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01), December 2001.
[32] M. Vincze, Robust tracking of ellipses at frame rate, Journal of the Pattern Recognition Society 34 (2) (2001) 487–498.
[33] S. Wesolkowski, Color image edge detection and segmentation: a comparison of the vector angle and the Euclidean distance color similarity measures, Master's Thesis, Systems Design Engineering, University of Waterloo, Canada, 1999.
[34] W.J. Wilson, C.C.W. Hulls, G.S. Bell, Relative end-effector control using Cartesian position-based visual servoing, in: Proceedings of the IEEE Transactions on Robotics and Automation 12:5, October 1996, pp. 684–696.
[35] B.H. Yoshimi, P.K. Allen, Active, uncalibrated visual servoing, in: Proceedings of the IEEE International Conference on Robotics and Automation, 1994, pp. 156–161.
[36] H. Zhang, J.P. Ostrowski, Visual servoing with dynamics: control of an unmanned blimp, in: Proceedings of the IEEE International Conference on Robotics and Automation, 1999, pp. 618–623.
[37] M. D'Zmura, P. Lennie, Mechanisms of color constancy, Journal of the Optical Society of America A 3 (10) (1986) 1662–1672.

Geert De Cubber was born in Halle, Belgium, in 1979. He received his master in electromechanical engineering at the Vrije Universiteit Brussels (VUB), Belgium, in 2001. He is currently pursuing a Ph.D. degree at the Department of Electronics and Information Processing (ETRO), VUB, Belgium. His research interest goes out to computer vision algorithms which can be used by intelligent mobile robots.

Sid Ahmed Berrabah received the degree in electronics and systems' control engineering as well as the master degree in signals and systems from Tlemcen University, Algeria, in 1994 and 1996, respectively. He is an assistant professor at Tlemcen University since 1997. Currently he is pursuing the Ph.D. degree at the Department of Electronics and Information Processing (ETRO), VUB, Belgium. His research interests include motion analysis and map building for mobile robot navigation.

Hichem Sahli was born in Tunis, Tunisia, in 1960. He is currently a professor of image analysis and computer vision at the Vrije Universiteit Brussel, Department of Electronics and Information Processing, Brussels, Belgium. He coordinates the research team in computer vision. His research interests include image analysis and interpretation, computer vision, mathematical morphology, scale-space theory, image registration, image sequence analysis, multispectral image processing and 3D reconstruction.