

Vision for Interaction

H. I. Christensen, D. Kragic, and F. Sandberg

Centre for Autonomous Systems
Numerical Analysis and Computer Science
Royal Institute of Technology
SE-100 44 Stockholm, Sweden

1 Introduction

Society is experiencing a significant aging over the next few decades [1]. This will result in an increase by 30% more elderly and retired people and an increase of 100% in the number of people above 85 years of age. This increase in age will require significant new services for managed care and new facilities for providing assistance to people in their homes to maintain a reasonable quality of life for society in general and elderly and handicapped in particular. There are several possible solutions to the aging problem and the delivery of the needed services. One of the potential solutions is use of robotic appliances to provide services such as cleaning, getting dressed, mobility assistance, etc. In addition to providing assistance to elderly it can further be envisaged that such robotic appliances will be of general utility to humans both at the workplace and in their homes, for many different functions.

At the same time the basic methods for navigation and operation in a domestic environment is gradually becoming available. Today there are methods available for (semi-) autonomous mapping and navigation in domestic settings [2, 3]. For service robots to be truly useful they must include facilities for interaction with the environment, to be able to pick-up objects, change controls, support humans, etc. Interaction can be implemented using dedicated actuator systems such as lift and simple grippers as found on many research platforms. The ultimate facility for interaction is of course a light-weight mobile manipulator that can support a range of different tasks.

Interaction with objects requires facilities for recognition of objects, grasping and manipulation. For the instruction of a robot to carry out such tasks there is also a need for flexible interaction with human users. The by far most flexible sensory modality that provides methods for both recognition, action interpretation, and servoing is computational vision. In this paper the issues of computational vision for human-computer interaction and visual servoing for manipulation will be discussed and example solutions for use of vision in the context of the service robot systems: The Intelligent Service Robot (ISR) [4] will be presented. The ISR system is a service robot demonstrator aimed at operation in a natural domestic setting for fetch and carry type tasks. The system is to be used by regular people for operation in an unmodified setting, which implies that it must rely on sensory information for navigation, interaction and instructions. It is in general assumed that the users have no special training which imposes special requirements in terms of interaction.

The paper is organised in the following manner: In Section 2 the use of vision for gesture interpretation is presented, Section 3 presents a systems for visual servoing for object manipulation, while Section 4 provides a summary and directions for future research.

2 Human-Robot Interaction

2.1 Structure of Interaction System

Interaction with humans can take on a number of different forms. The most typical forms are outlined below in terms of output and input;

Output

Screen Output in terms of text or graphics on a computer screen is quite common for human computer interaction and widely studied in HCI. Mounting a computer screen on the robot requires that the user is situated in front of the screen, which is a rather strong requirement.

Speech Spoken feedback is becoming more and more common as the quality of speech synthesis is improving. Excellent examples includes the Festival systems from University of Edinburgh and commercial systems from Microsoft and IBM. The advantage of spoken feedback is that it can be made omni directional and thus gives added flexibility in terms of position with respect to the robot. Unfortunately the spoken feedback might be challenging for elderly (with reduced hearing) and in noisy environments.

Embodied agent An alternative to graphics and speech is use of an embodied agent, where a small character is used for feedback. An example of such an agent is show in figure 1. The agent has four degrees of freedom. The head can perform pan and tilt movements, corresponding to nodding and shaking of the head. Each of the arms have a single degree of rotational freedom, i.e. lifting of arms. Using this methodology it is possible to nod the head as a “yes” response and shake the head as a “no” response. In addition the agent can raise both hands to indicate surrender, i.e. giving up as part of a dialogue (“I did not understand what you said”). This is an intuitive modality for feedback and provides a good complement to speech and/or graphics.

Input

Keyboard Keyboard and mouse are well known modalities for interaction with GUIs but in the context of a mobile robot it is not immediately obvious where to put the keyboard and/or the mouse. Wireless keyboards are available, but one would have to carry such a unit around. This modality is thus not very useful for mobile platforms.

Touch screen Some commercial systems such as the HelpMate use touch screens for interaction with users. This is a flexible way of providing input, provided that all possible commend alternatives can be enumerated on the screen. Unfortunately the modality requires that the user is situated in front of the screen to enable interaction.



Fig. 1. Example of an embodied agent (Developed by the Industrial Designer: Erik Espmark)

Speech Input Gradually speech interpretation systems with large vocabularies are becoming available. Good examples include the IBM ViaVoice and Dragon Dictate. Both of these systems are unfortunately speaker specific and require extensive training before they can be deployed. For limited vocabularies it is now possible to use speaker independent software for input generation. An excellent example is the Esmeralda system developed by Fink et al. [5, 6]. A problem with speech systems is that the signal to noise ratio typically must be very good to obtain reasonable recognition rates. This either requires a good microphone in combination with a quiet setting or alternatively mounting of a microphone on the user (i.e. a wireless microphone). The second option is by far the most frequently used option. Use of specific vocabularies is well suited for specification of a sequence of commands such as “fetch the milk from the refrigerator”. Unfortunately such a mode of interaction is only well suited for a set of prior known objects that can be uniquely identified. Thus a system would not be able to understand commands such as “pick this object and deliver it to Susan”, or “please clean this area”. Use of speech along is thus only suitable for well defined settings.

Gestures Gestures are well suited for generation of instructions in terms of a specific set of commands. The sequencing of commands will typically require recognition of detailed timing which in most cases is difficult if not impossible. In addition a problem with gestures is that the user has to stand in front of the robot cameras and the action must be observable by the camera. This is a significant constraint. In addition vision has often had robustness problems for operation in domestic settings around the clock. Gestures are however well suited for specification of spatial locations and for simple instructions such as “follow me”, “this area”, “go left”, “STOP!”, etc.

Given the characteristics outlined one possible solution is to use a spoken dialogue systems in combination with a simple set of gestures and an embodied agent for basic level feedback. For processing of spoken input the Esmeralda system [6] is used. The spoken system provides a sequence of commands, while the gesture system may be

used for generation of motion commands in combination with the speech systems. In the following the basic components of the gesture interpretation system are outlined.

2.2 Gesture Interpretation

Gestures have been widely studied in computer vision and human computer interaction as for example reported by [7, 8]. Gestures have also been used in a few robot applications as for example reported by [9, 10].

For interaction with a human operator it is necessary to consider the location of hands and the face of the users so as to allow both left and right handed people to interact with the system. In addition for some commands it is of interest to be able to use both hands. There is thus a need for the definition of a system that allows identification of the face and the two hands. Once the regions corresponding to these three components have been identified they must be tracked over time and finally the trajectories must be interpreted to allow recognition of the specific gestures. The process of gesture interpretation can thus be divided into three steps i) segmentation of hands and face, ii) tracking of regions and iii) interpretation of trajectories. Each of the three processes are described in detail below.

Colour segmentation in chromaticity space Recognition of hands and other skin coloured regions have been studied extensively in the literature in particular for face and gesture recognition. The by far most frequently used approach is based on colour segmentation. To achieve robustness to colour variation it is well known that the traditional RGB colour representation is not a good basis for segmentation. Frequently researchers such as Sterner [8] have used a Hue-Saturation-Intensity (HSI) representation as a basis for recognition. In this colour space Hue and Saturation are used for segmentation while the intensity component is ignored. Such approaches allow definition of methods that are more robust to light variation. Unfortunately not only intensity but also colour temperature varies over the day and is also dependent on the use of natural or artificial lighting. It is thus easy to illustrate that HSI based approaches will fail if the colour temperature is varied. A more careful consideration of the underlying physics reveals that the reflected light from an object such as the human skin can be modeled using a di-chromatic reflection model, i.e.:

$$L(\theta, \lambda) = m_s(\theta)L_s(\lambda) + m_bL_b(\lambda) \quad (1)$$

where L is the light reflected as a function angle of incidence (θ) and the wavelength (λ). The reflected light is composed of two components one derived from surface reflection (L_s) and another dependent on light reflected from below the surface (L_b). The skin colour and the content of melanin will determine the reflection coefficients (m_s and m_b). The total light reflected can be determined through integration over the full spectrum, using a spectral sensitivity function for the camera of $f_{RGB}(\lambda)$, i.e.:

$$C_{RGB} = \int_{\lambda} L(\theta, \lambda) f_{RGB}(\lambda) d\lambda \quad (2)$$

The colour image is unfortunately still highly sensitive to illumination variations. It is however possible to change the image to Chromaticity coordinates where a more robust representation can be achieved. I.e.,

$$\begin{aligned} r &= \frac{R}{R + G + B} \\ g &= \frac{G}{R + G + B} \\ b &= \frac{B}{R + G + B} \end{aligned}$$

When processing images in chromaticity space normally only the $r - g$ components are used. The b component is redundant and in addition normally the blue channel of a camera is noisy in particular in low light conditions.

When considering images of human skin in $r - g$ space it is possible to identify a narrow region that contains all skin pixels. This region is often termed the skin locus [11]. Through simple box classification in $r - g$ space it is possible to perform a segmentation of skin regions with a high level of confidence. An example is shown in figure 2.

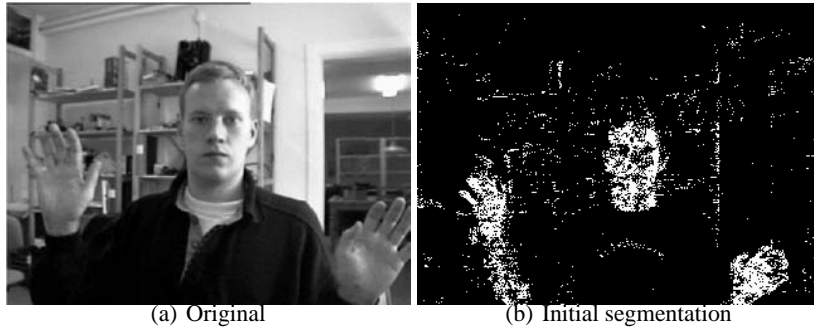


Fig. 2. Initial segmentation of skin colour regions

Through estimation of the density of skin coloured pixels in a local neighbourhood and subsequent thresholding it is possible to detect the three major skin coloured regions in an image. For images where a single person is the dominating figure this will result in reliable detection of the face and hands, as shown in figure 3:

To further compensate for illumination variations it is possible to compute a histogram over the detected regions (in $r - g$ space) and feedback the histogram boundaries back to the estimation of the location of the skin locus box classifier to allow adaptation to potential variations in illumination.

Region Tracking Once regions have been extracted the next problem is to track the regions over time to allow generation of a sequence of trajectories for interpretation of

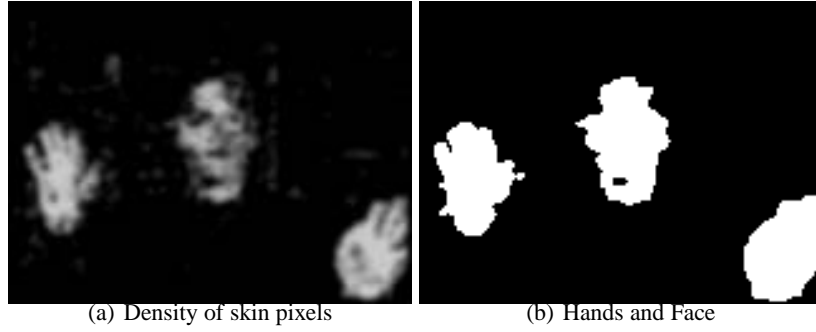


Fig. 3. Estimation of density of skin pixels and thresholding of the density image to extract hand and face regions

gestures. The tracking is here carried out using a conventional Kalman filter [12]. The regions are tracked directly in image space. For the tracking the center of gravity is used, i.e:

$$\mathbf{p} = \begin{pmatrix} x \\ y \end{pmatrix} \quad (3)$$

For the tracking the position \mathbf{p} and velocity $\dot{\mathbf{p}}$ is used. This results in a system model, where the state is:

$$\mathbf{x} = \begin{pmatrix} \mathbf{p} \\ \dot{\mathbf{p}} \end{pmatrix} \quad (4)$$

Under the assumption of a first order Taylor expansion the autonomy of the system is defined by:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{C}w_k \quad (5)$$

$$\mathbf{A} = \begin{pmatrix} 1 & \Delta T \\ 0 & 1 \end{pmatrix} \quad (6)$$

$$\mathbf{C} = \begin{pmatrix} \Delta T^2/2 \\ \Delta T \end{pmatrix} \quad (7)$$

Where w_k models acceleration etc and is assumed to be Gaussian noise, with a variance σ_w^2 . As only the position of regions are available in the image, the observation model is defined as:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x} + n_k \quad (8)$$

$$\mathbf{H} = \begin{pmatrix} 1 & 0 \end{pmatrix} \quad (9)$$

Where \mathbf{z} is the measurement vector and n_k is the measurement noise that is assumed to Gaussian noise with a variance σ_n^2 . Using the standard Kalman updating model it is now possible to track regions over time. Details can be found in [12]. The matching between images is performed using a nearest neighbour algorithm, which is adequate when the algorithm is run at 25 Hz.

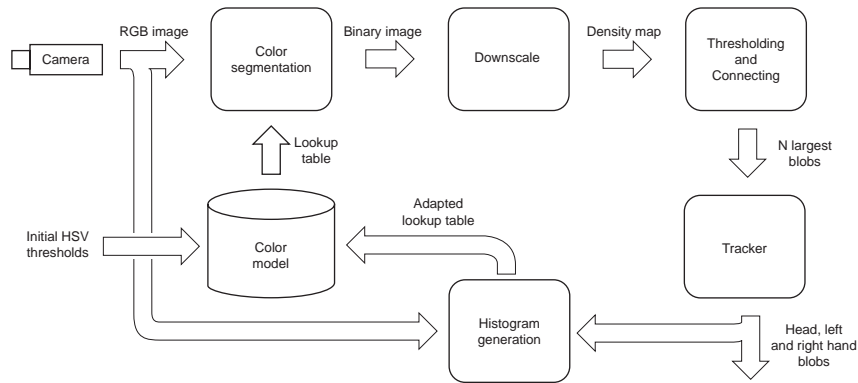


Fig. 4. The system for face and head region tracking

The full systems for tracking of hand and face regions is shown in figure 4.

An example result for tracking of regions is shown in figure 5. The trajectories illustrate how both hands and the face are tracked over a limited sequence.

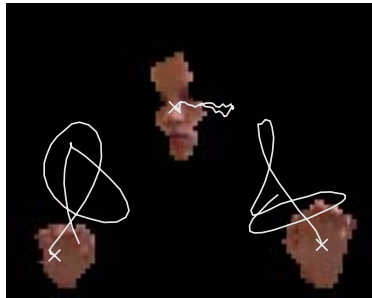


Fig. 5. Example trajectories extracted for face and hand tracking

The above mentioned system has been implemented on a standard 400 MHz Pentium-II computer running Linux. The system uses a standard Matrox frame-grabber. The system is running in real-time. The execution time for the different phases of the algorithm are shown in Table 1 (for an 160x120 pixel image):

Interpretation of gestures using HMM From the tracking module three sets of trajectories are available. These trajectories can be used for the interpretation of gestures. For this purpose it is assumed that there is some minimum motion between different gestures. I.e. the hands are almost stationary between different gestures. Under this assumption velocity can be used to segment the trajectories into independent parts that can be processed independent of each other. Once separate trajectories have been ex-

Phase	Time/Frame [ms]
Image Retrieval	4.3
Colour Segmentation	0.6
Density Estimation	2.1
Connected Components	2.1
Kalman filtering	0.3
Total	9.4

Table 1. CPU usage for segmentation and tracking of hand and face regions on a 400 MHz Pentium II computer running Linux

tracted they are normalised to provide independence of size and position in the image. The normalized trajectory segments are forwarded to a recognition module.

For the recognition of gestures a Hidden Markov Model is used [13]. A Hidden Markov Model (HMM) is defined by a 5-tuple: $\lambda = (S, V, A, B, \pi)$ that is defined as:

- States, $S = \{s_1, s_2, \dots, s_N\}$. The state at time t is denoted q_t .
- Symbols, $V = \{v_1, v_2, \dots, v_m\}$
- Transition probability distribution, $A = \{a_{ij}\}$, where
 $a_{ij} = P(q_{t+1} = s_j | q_t = s_i), i, j \in \{1..N\}$
- Observation probability distribution, $B = \{b_j(k)\}$, where
 $b_j(k) = P(v_k \text{ generated in state } j), j \in \{1..N\}, k \in \{1..M\}$
- Initial state distribution, $\pi = \{\pi_i\}$ where
 $\pi_i = P(q_i = s_i), i \in \{1..N\}$.

In this particular application the HMM is assumed to be a Markov chain, and all trajectories are assumed to start in the first state, i.e.

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1 & i = 1 \end{cases} \quad (10)$$

In many applications the observations are organised into a sequence, i.e. $O = \{x_1, x_2, \dots, x_K\}$. In this particular application the gestures are continuous rather than discrete and the observation probability distribution is thus not discrete, and is consequently replaced by a continuous Gaussian distribution:

$$b_j(x_t) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_j|}} e^{-\frac{1}{2}(x_t - \mu_j)^T \Sigma_j^{-1} (x_t - \mu_j)} \quad (11)$$

where n is the dimension of the measurement space (here 2) and μ is the mean and Σ_j is the covariance matrix for the observations in state j .

For the use of a HMM two problems have to be considered:

1. *Evaluation*: Determining $P(O|\lambda)$, the probability that a HMM λ generated a particular observation sequence. I.e. is a HMM for a particular gesture the best model for the observed trajectory.
2. *Training*: Given a sequence of test samples how does one determine the HMM that best “explains” the data set.

Let us consider each of these problem in the following.

Recognition/Evaluation: Recognition is performed by using the HMM that is the most like generator of a particular sequence. A simple algorithm to compute $P(O|\lambda)$ is through use of the Vieterbi algorithm [13]. Given:

$$\alpha_i(t) = P(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t, q_t = s_i | t)$$

Initialize:

$$\alpha_i(1) = \pi_i b_i(\mathbf{x}_1), i \in \{1..N\} \quad (12)$$

Recursion

$$\alpha_j(t+1) = \left[\max_i \alpha_i(t) a_{ij} \right] b_j(\mathbf{x}_{t+1}), t \in \{1..T-1\}, j \in \{1..N\} \quad (13)$$

Termination

$$P(O|\lambda) = \max_i \alpha_i(K), i \in \{1..N\} \quad (14)$$

The Vieterbi algorithm generates a maximum likelihood estimate for the trajectory. To allow identification of the case “no gesture” it is necessary to introduce a threshold that ignores low probability trajectories.

Training The training phase is carried out by using a sequence of test trajectories that are recorded and subsequently processed in a batch process. The training is carried out in two phases: i) initialisation and ii) re-estimation.

The initialisation of the HMM is carried out by simple statistics, i.e.

$$\boldsymbol{\mu}_i = \frac{1}{N_j} \sum_{\forall t, q_i = s_j} \mathbf{x}_t \quad (15)$$

and

$$\Sigma_j = \frac{1}{N_j} \sum_{\forall t, q_i = s_j} (\mathbf{x}_t - \boldsymbol{\mu}_j)(\mathbf{x}_t - \boldsymbol{\mu}_j)^T \quad (16)$$

where N_j is the number of observations in state j . The most likely trajectories are determined using the Vieterbi algorithm, and the state transition probabilities are approximated by relative frequencies:

$$a_{ij} = \frac{A_{ij}}{\sum_{k=1}^N A_{ik}} \quad (17)$$

The initialisation process is repeated until the estimates stabilize.

The initial parameters can be improved through use of the Baum-Welch forward-backward algorithm [14]. Let $L_j(t)$ denote the likelihood of being in state j at time t . Then the statistics can be revised to be:

$$\boldsymbol{\mu}_j = \frac{\sum_{t=1}^T L_j(t) \mathbf{x}_t}{\sum_{t=1}^T L_j(t)} \quad (18)$$

and

$$\Sigma_j = \frac{\sum_{t=1}^T L_j(t) (\mathbf{x}_t - \boldsymbol{\mu}_j)(\mathbf{x}_t - \boldsymbol{\mu}_j)^T}{\sum_{t=1}^T L_j(t)} \quad (19)$$

The calculation of the state likelihoods, $L_j(t)$, is carried out using the forward-backward algorithm. The forward probability is defined by

$$\alpha_i(t) = P(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t, q_t = s_i | \lambda) \quad (20)$$

Computed as mentioned in the recognition section (i.e., Eqs 12 – 14). The backward probability is defined as

$$\beta_i(t) = P(\mathbf{x}_{t+1}, \mathbf{x}_{t+2}, \dots, \mathbf{x}_K | q(t) = i, \lambda) \quad (21)$$

The forward backward probabilities gives:

$$\alpha_i(t)\beta_i(t) = P(O, q_t = i | \lambda) \quad (22)$$

Which can be used for computation of the state probabilities and the state transition probabilities.

$$L_j(t) = \frac{1}{P(O|\lambda)} \alpha_j(t)\beta_j(t) \quad (23)$$

$$a_{ij} = \frac{\frac{1}{P(O|\lambda)} \sum_{t=1}^{T-1} \alpha_i(t)b_j(\mathbf{x}_{t+1})\beta_j(t+1)}{\frac{1}{P(O|\lambda)} \sum_{t=1}^{T-1} \alpha_i(t)\beta_j(t)} \quad (24)$$

The forward backward process is then repeated until the probabilities stabilize by which the network is ready for application.

The HMM classifier has been implemented on the same PC as used for the blob tracking. The classifier run in 23.4 ms for each full gesture and combined the blob tracking and the recognizer is thus able to classify gestures in real-time.

Evaluation The gesture system has been evaluated using two different gesture sets. For the initial evaluation of the system the graffiti language developed by Palm Computing was implemented. The graffiti language has without doubt been developed to ensure a minimum of errors in recognition and consequently it is a good dataset for testing of the algorithm. After quite some initial testing a Markov model with four states was selected as a model.



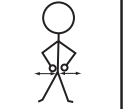
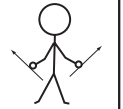

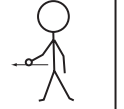
For the testing a total of 2230 test sequences were recorded. The dataset involved 5 different persons all of them operating in a living room scenario. From the dataset 1115 of the sequences were used for training of the HMM. Subsequently the remaining 1115 sequences were used for testing of the algorithm. For recognition tests were carried using image position \mathbf{p} , image velocity $\dot{\mathbf{p}}$ and position and velocity. The achieved recognition rates are summarised in table 2.

Subsequently a language suitable for simple instruction was designed to allow for interaction with the robot. The basic instructions are feed into the robot control system to allow real-time interaction. The set of gestures is shown in figure 6.

The gesture set was trained on 500 image sequences. Subsequently the gestures were tested on a new set of 4 people using a limited set of 320 sequences. A recognition rate of 78% was achieved for this dataset. The lower recognition rate is in part due to

Recognition rates	Position	Velocity	Combined
Result [%]	96.6	87.7	99.5

Table 2. Recognition results achieved for the Palm graffiti language

					
Attention	Idle	Forward	Back	Left	Right






				
Turn left	Turn right	Faster	Slower	Stop

Fig. 6. The gesture set used for basic robot interaction

the fact that the gesture set has not been optimized. Another problem experienced with a data set that involves both hands is occasional occlusions between the hands, which easily can lead to errors in the tracking process. When combined with the speech system it is however possible to obtain an overall system for human interaction that is suitable for use by regular people.

3 Model Based Visual Manipulation

One of the basic requirements of a service robot is the ability to manipulate objects in a domestic environment. Given a task such as to fetch a package of milk from a refrigerator, the robot should safely navigate to the kitchen, open the refrigerator door, fetch and deliver the milk. Assuming a perfect positioning in front of the refrigerator door, two tasks remain: opening the door and fetching the milk. The first task - opening of the door may benefit from integrated use of different sensors, a camera and a force-torque sensor. This problem has been addressed in our previous work, [15]. Here, we concentrate on “fetch-and-carry” tasks.

A “fetch” or a “pick-up” task where vision is used as an underlying sensor may in general be divided into the following subtasks:

Detection Given a view of the scene, the agent should be able to provide a binary answer whether the desired object is in the field of view or not. This implies that the scene might both contain a few similar objects or no object at all. “Simple” approaches based on colour or texture may be used for moderately complex scenes. However, it is not straightforward to design a general tool for highly dynamic environments where spatial position of objects may change completely unexpectedly.

Segmentation Providing that the detection was successful, the image must be searched for the object. The hypotheses should be made and the image has to be segmented for further verification and actual recognition of the object. In a trivial manner, an opposite approach may be used where we exclude those regions that are not likely to contain the object, e.g. regions of uniform colour or highly textures region (depending, of course, on the objects appearance).

Recognition The recognition tool should recognize the object and provide certainty measure for the recognition. Depending on the implementation, this part may also provide partial/full pose of the object [16]. Object recognition is a long studied research issue in the field of computational vision [17]. A promising system, based on Support Vector Machines [18] has recently been developed locally and used to detect a moderate set of everyday objects.

Alignment After the object has been located, the robotic manipulator/arm should be visually servoed to the vicinity of the object. Many of the existing visual servo systems neglect the first three issues and concentrate on the alignment task. The approaches differ in the number of cameras used (stand-alone vs. eye-in-hand) and underlying control space (image based [19], position based [20] and 2 1/2 D visual servoing [21]). To perform the alignment in a closed-loop manner, the systems should have the ability to track object features during the arm movements and update their position if needed. This is especially obvious if the objects are not static or if the alignment and grasping are performed while the robot is moving.

Grasping After the alignment has been performed, the grasping sequence should start. For “simple” objects like boxes and cylinders (which most of the food items have) a set of predefined grasps may be learned and used depending on the current pose of the object. However, there are many everyday object which are far from having a “nice”, easy-graspable shape. It is obvious that for those objects vision should be used together with other sensory modalities like force-torque or touch sensors.

Manipulation In many cases the agent is required to deliver an object in a particular way, i.e. the object should be fit into or put onto some other object. This may additionally require of the agent to manipulate the object in the end-effector/hand before final delivery/placement. There therefore a need form the grasping tool to perform both a stable and a manipulable grasp, [22], [23] .

Each of the outlined issues have been widely studied in different research areas: computer vision, control, optimization, sensor integration, etc. However, their interaction and integration is still quite a young area.

Although somewhat neglected because of it computational complexity, the use of CAD models has been a quite popular approach to solving all of the outlined issues: pose estimation and tracking, [24, 25], alignment [26, 27] and grasping [28, 22]. This is also the approach taken in our current work. In this paper we address the issues of pose estimation, alignment (visual servoing) and tracking. For more information about the currently used recognition tool we refer to [18]. The diagram of the implemented tracking system is presented in figure 7. The approach is similar to the one proposed by Marchand in [26].

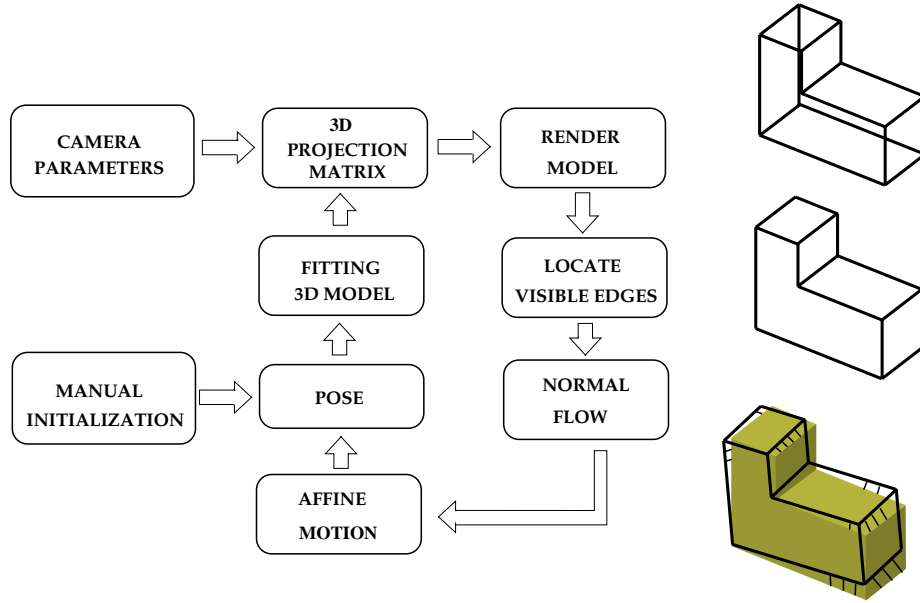


Fig. 7. Diagram of the model based tracking system.

3.1 Pose Estimation

Pose estimation considers a computation of a rotation matrix (orientation) and a translation vector of the object (position), $\mathbf{H}(\mathbf{R}, \mathbf{t})$. Different researchers have formulated closed form solutions to pose estimation with feature points in coplanar or non-coplanar configuration,[29–33]. It has been shown that the key to a robust estimation is the use a larger number of feature points since the image noise and measurement errors average out due to the feature redundancy. The most straightforward method described by Roberts, [29], consists in retrieving the 11 parameters of a projective transformation matrix as a solution to a linear system. Other notable methods were proposed by Tsai, [34], Lowe [24], and Yuan, [35]. Mentioned techniques rely on the Newton-Raphson method which requires the initial approximate pose as well as the computation of the Jacobian matrix which is computationally expensive and usually not suitable for real-time applications.

The method proposed by DeMenthon and Davis[36], relies on linear algebra techniques. Although this method is also iterative, it does not require the *a-priori* knowledge of the initial pose nor the estimation of the Jacobian matrix. At the first iteration step, the method computes the pose from orthography and scaling with the assumption that the image projections of model points were obtained under a scaled orthographic projection. Briefly, the method starts by assuming a scaled orthographic projection and iteratively converges to a perspective projection by minimizing the error between the original image points and projected point using the current camera model. The method converges after 3-4 iterations which is suitable for real-time applications. In our im-

plementation, this step is followed by an extension of Lowe's [24] nonlinear approach proposed in [37]. This step is called POSE in figure 7.

Once the pose of the object in the camera coordinate system is known, the tracking is initiated. It involves 3 steps:

1. normal flow computation
2. fitting an affine/quadratic transformation model between two views
3. optimizing the pose with respect to spatial intensity gradients in the image

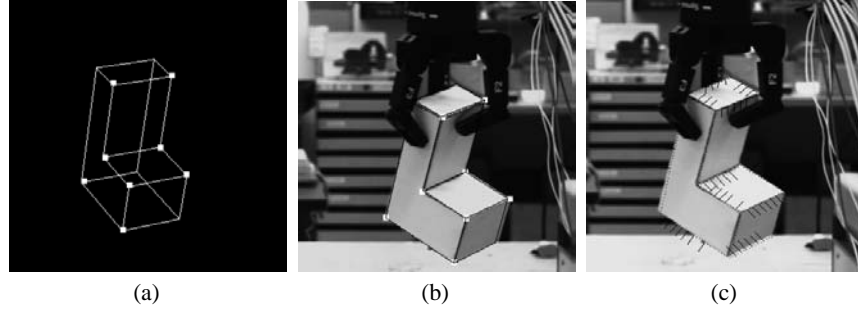


Fig. 8. Example images obtained during pose estimation and tracking: a) the model with points used in POSE to determine the pose, b) pose estimation (the model is overlaid in black), and c) normal flow estimation (the lines represent the direction and not the magnitude).

3.2 Model Based Tracking

After the pose estimation is obtained, the internal camera parameters are used to project the model of the object onto the image plane. A simple rendering technique is used to determine the visible edges of the object [38]. For each visible edge, tracking nodes are assigned at regular intervals in image coordinates along the edge direction. After that, a search is performed for the maximum discontinuity (nearby edge) in the intensity gradient along the normal direction to the edge. The edge normal is approximated with four directions: 0, 45, 90, 135 degrees. This way we obtain a displacement vector:

$$\mathbf{d}_i^\perp = \begin{pmatrix} \Delta x_i \\ \Delta y_i \end{pmatrix} \quad (25)$$

representing the normal displacement field of visible edges.

A 2D affine transformation is expressed by:

$$\begin{pmatrix} x_i^{t+1} \\ y_i^{t+1} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \begin{pmatrix} x_i^t \\ y_i^t \end{pmatrix} + \begin{pmatrix} T_x \\ T_y \end{pmatrix} = \mathbf{W}(\mathbf{p}_i)\Theta \quad (26)$$

with $\Theta = (a_1, a_2, a_3, a_4, T_x, T_y)^T$, $\mathbf{p}_i^t = (x_i^t, y_i^t)^T$, $\mathbf{p}_i^{t+1} = \Psi_\Theta \mathbf{p}_i^t$ (where Ψ_Θ denotes affine transformation from (Eq. 26)) and

$$\mathbf{W}(\mathbf{p}) = \begin{pmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{pmatrix} \quad (27)$$

Displacement vector can be written as:

$$\mathbf{d}_i(\mathbf{p}_i) = \mathbf{W}(\mathbf{p}_i)\Theta' \quad (28)$$

$$= \mathbf{W}(\mathbf{p}_i) (\Theta - (1, 0, 0, 1, 0, 0)^T) \quad (29)$$

From $(\mathbf{p}_i^t, \mathbf{d}_i)_{i=1\dots k}$, we estimate the 2D affine transformation Θ' . From Eq. 29, we have:

$$\mathbf{d}_i^\perp = \mathbf{n}_i^T \mathbf{d}_i(\mathbf{p}_i) = \mathbf{n}_i^T \mathbf{W}(\mathbf{p}_i)\Theta' \quad (30)$$

where \mathbf{n}_i is a unit vector orthogonal to the edge at a point \mathbf{p}_i . From Eq. 30 we can estimate the parameters of the affine model, $\hat{\Theta}'$ using a M-estimator ρ :

$$\hat{\Theta}' = \arg \min_{\Theta'} \sum_i \rho(\mathbf{d}_i^\perp - \mathbf{n}_i^T \mathbf{W}(\mathbf{p}_i)\Theta') \quad (31)$$

Computed affine parameters give us new image positions of the points in time $t + 1$. Thereafter, pose estimation step (POSE) is performed to obtain the pose of the object in the camera coordinate system, $\mathbf{H}(\mathbf{R}, \mathbf{t})_{init}$.

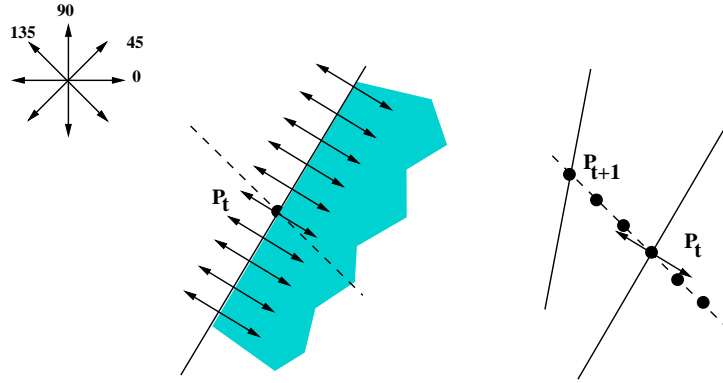


Fig. 9. Determining normal displacements for points on a contour in consecutive frames.

In certain cases, a six-parameter motion model is not sufficient. The following polynomial model may be used:

$$\begin{pmatrix} x_i^{t+1} \\ y_i^{t+1} \end{pmatrix} = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} + \begin{pmatrix} a_2 & a_3 \\ a_4 & a_5 \end{pmatrix} \begin{pmatrix} x_i^t \\ y_i^t \end{pmatrix} + \begin{pmatrix} a_6 & a_7 & 0 \\ 0 & a_6 & a_7 \end{pmatrix} \begin{pmatrix} x_i^{t^2} \\ x_i^t y_i^t \\ y_i^{t^2} \end{pmatrix} = \mathbf{W}(\mathbf{p}_i)\Theta \quad (32)$$

Fitting the CAD model to the spatial intensity gradients

Using the estimated affine parameters $\hat{\Theta}'$ and positions of edge nodes at time t , we are able to compute their positions at time $t + 1$ from:

$$\mathbf{p}_i^{t+1} = \Psi_{\Theta} \mathbf{p}_i^t \quad (33)$$

with

$$\hat{\Theta} = \hat{\Theta}' + (1, 0, 0, 1, 0, 0)^T \quad (34)$$

As already mentioned, the affine motion model does not completely account for the 3D motion and perspective effects that occur during the object motion. Therefore, the pose space, $\mathbf{H}(\mathbf{R}, \mathbf{t})$, should be searched for a best fit given the image data. As proposed in [26], the projection of the object model is fitted on the spatial intensity gradients in the image, using the $\mathbf{H}(\mathbf{R}, \mathbf{t})_{init}$ as the initialization:

$$\hat{\mathbf{H}}(\mathbf{R}, \mathbf{t}) = \arg \min_{\mathbf{H}} \left[- \sum_{C_{\mathbf{H}}} \|\nabla \mathbf{G}(t+1)\| \right] \quad (35)$$

where $\nabla \mathbf{G}(t+1)$ is the intensity gradient along the projected model edges and $C_{\mathbf{H}}$ are the visible edges of the model given a pose $\mathbf{H}(\mathbf{R}, \mathbf{t})$.

The optimization method proposed in [26] uses a discrete hierarchical search approach with respect to the pose parameters. However, the right discrete step of pose parameters is crucial in order to find the right minimum value. The affine motion model is particularly sensitive to the rotational motions of the object and large changes in rotation usually result with loss of tracking. For that reason we have extended the method proposed in [26] so that the step determination is dynamically changed based on the 3D object velocity.

This system has been used for three different tasks:

1. **Alignment and tracking** Position based visual servoing approach [39] is used to align the robot hand with the object. In addition, the robot hand follows the object during its motion by keeping the constant pose between the objects and hands coordinate systems.
2. **Grasping** Grasping is performed by using a set of pre-defined grasps depending on the objects pose.
3. **Visual servoing** After the object is grasped, image based visual servoing [39] is used to guide the manipulator during the placement task.

To control the robot we have adopted both image based and position based visual servo approaches. Detailed description and characteristics of these approaches can be found in [39]. The basic idea is to minimize an error function usually defined as an image position or 3D pose difference between the current and desired objects set of tracked features. Detailed description of the outlined tasks are presented in following section together with the experimental setup.

3.3 Experimental SetUp

The dextrous robot hand used in this example is the Barrett Hand. It is an eight-axis, three-fingered mechanical hand with each finger having two joints. One finger is stationary and the other two can spread synchronously up to 180 degrees about the palm (finger 3 is stationary and fingers 1 and 2 rotate about the palm). The hand is controlled by four motors and has 4 degrees of freedom: 1 for the spread angle of the fingers, and 3 for the angles of the proximal links. Our current model of the hand uses slightly simplified link geometries, but the kinematics of the model match the real hand exactly. The hand is attached to Puma560 arm which operates in a simple workcell. The vision system uses a standard CCD camera (Sony XC-77) with a focal length of 25mm and is calibrated with respect to the robot workspace. The camera is mounted on a tripod and views the robot and workcell from a 2m distance.

3.4 Alignment and Tracking

The objective of this experiment was to remain constant relative pose between the target and the robot hand. A typical need for such an application is during the grasp of a moving object or if the robot is performing a grasping task while moving. Position based visual servoing is used to minimize the error in pose between the hand coordinate system and a reference point defined in the object coordinate frame. When a stable tracking is achieved grasping may be performed.

3.5 Grasping

Here, pose estimation is used to perform a set of “learned” grasps. For each object we generate a set of reference points from which the grasp sequence should be initialized. These reference points depend on the pose of the object relative to the camera, i.e. the object might be placed horizontally, vertically, etc. First, we estimate the current pose of the object. After this, the manipulator is guided to the reference point that is either on a side (for a horizontal placement of the object) or above the object (if the object is placed vertically on the table). When the positioning is achieved, the “learned” grasp is performed.

3.6 Model Based Servoing

Here, the robot starts to move from an arbitrary position holding the grasped object. The task is to bring the object to some final or “thought” position where “teach by showing” approach was used [39].

The whole sequence consists of the following steps:

- The “teach by showing” approach is initiated in the following manner: the object is placed at some arbitrary position and its pose with respect to the camera coordinate system is estimated by matching points between the object and geometrical model using the approach presented in Section 3.1. Image coordinates of corner points are used to build a vector of desired image positions. After this, the object is placed to a new, initial position and tracking sequence is initiated. The vector of desired positions may also be known *a-priori*.

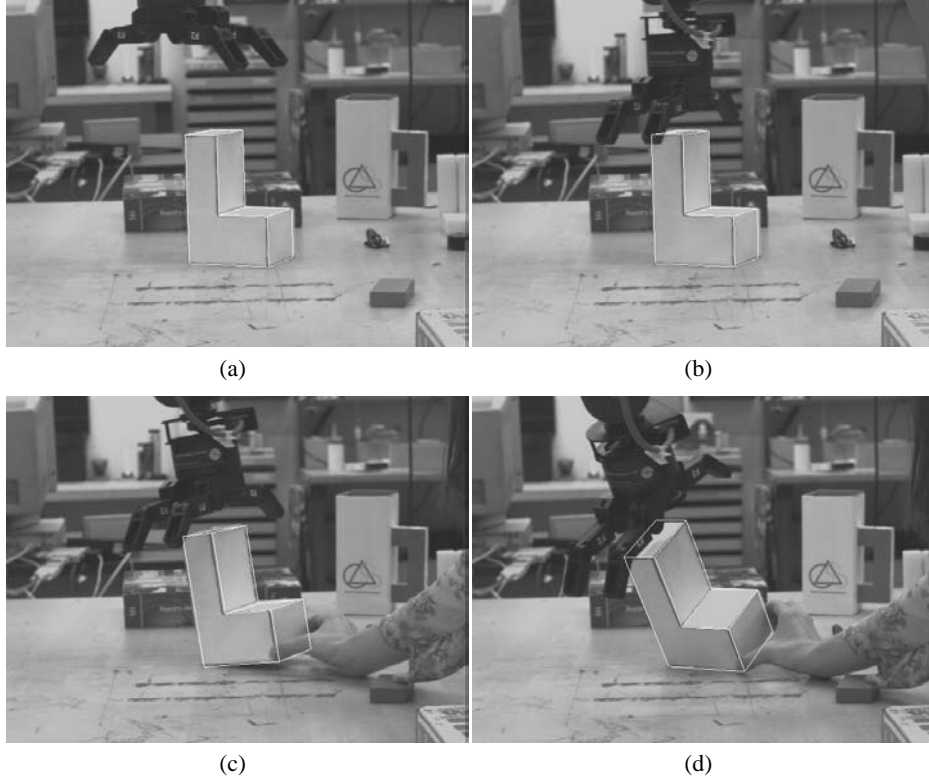


Fig. 10. From an arbitrary starting position in figure a), the end-effector is precisely servoed into a predefined reference position with respect to the target object, figure b). If the object starts moving, the visual system tracks the motion (pose) of the object and the robot is controlled keeping the constant relative pose between the hand and the object.

- In each frame, the current image positions of corner features are estimated using the technique presented in Section 3.2.
- The image Jacobian [39] is estimated and the control law obtained using a simple position controller.

4 Discussion

Vision is a highly flexible sensory modality for estimation of the state of the environment. This functionality can be used for construction of facilities for interaction with humans and objects for a variety of applications. In this paper it has been described how vision may be used for interpretation of gestures and how similar methods also may be used for recognition and servoing to facilitate grasping and manipulation. The presented methods have been evaluated in the context of an service robot application that operated in a domestic setting. A robust performance has been reported for the

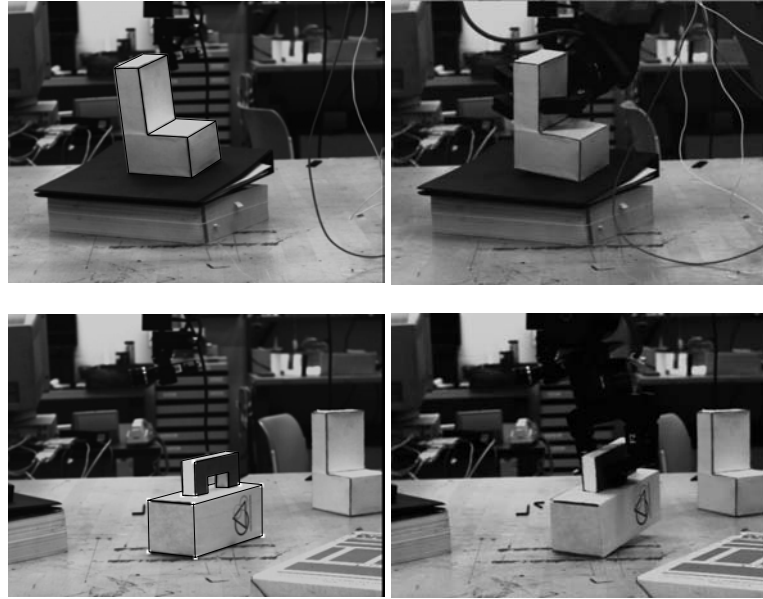


Fig. 11. Two examples of grasps: left) the vision system determines the pose of the object within the robot workspace and right) the completed grasp.

particular applications considered. The methods described provide basic facilities for interaction in a natural environment.

For interaction in natural environments it is necessary for the user to be able to provide fairly general commands and the robot must be able engage in a dialogue where turn-taking allow resolution of ambiguities and learning from past experience. So far the basic facilities for interaction has been provided but future research will need to consider how these methods can be integrated in a service robot application that can be used by regular people. To provide such an artifact there is a need for research on instruction by demonstration, which involves methods for dialogue design, task and behavioural modelling, and learning for generalisation etc. In addition the current methods allow recognition and interaction with prior defined objects. For operation in a general setting it is also necessary to provide methods that allow automatic acquisition of object models and grasping strategies. While the present paper has described a basic set of functionalities for a service robot there is still fundamental research to be carried out before service robots are ready to enter our homes.

Acknowledgment

This research has been sponsored by the Swedish Foundation for Strategic Research through its Center for Autonomous Systems. The support is gratefully acknowledged.

The research presented in this paper was carried out in the context of the Intelligent Service Robot project and as such some of the results are based on prior work by Patric

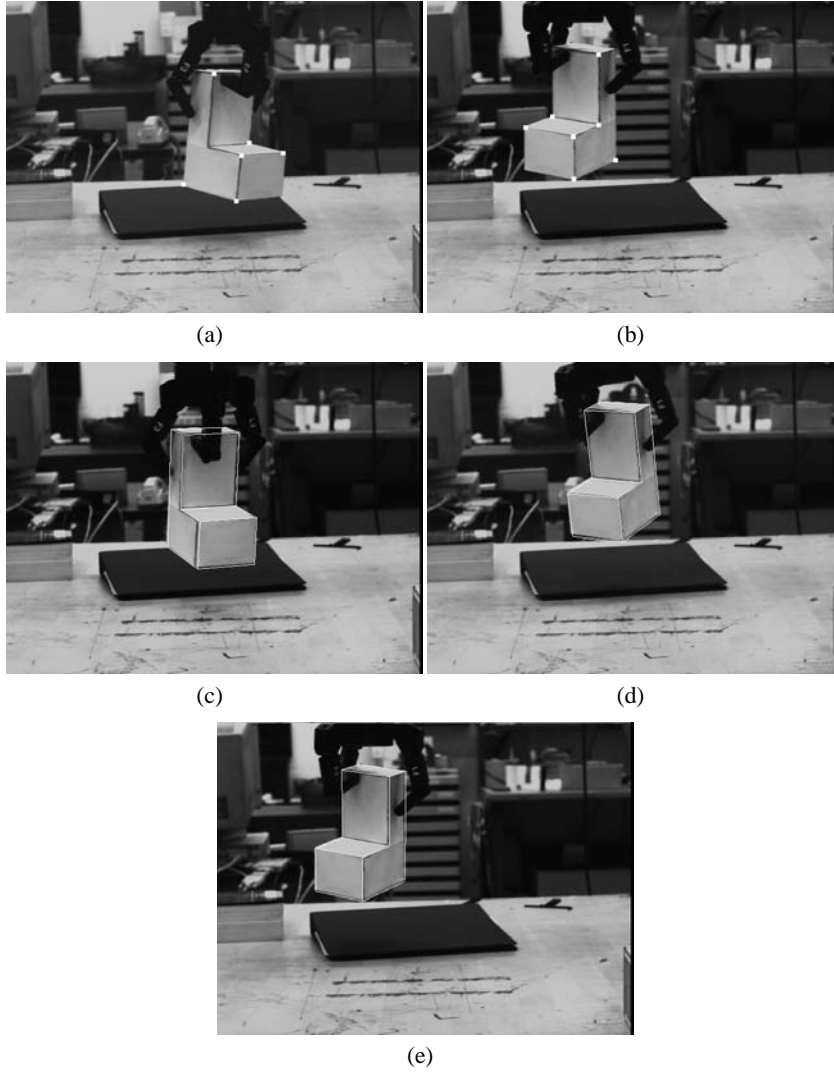


Fig. 12. a) Start pose, b) Destination pose, c), d), e) Intermediate images of visual servo sequence. This particular test was performed mainly in order to test the visual tracking system. The object undergoes a rotational motion around camera's y -axis. However, the tracking is remained during the whole servoing loop.

Jensfelt, Anders Orebeck, Olle Wijk, David Austin, Lars Petersson and Magnus Andersson. Without their assistance this work would not have been possible. Finally the work has benefited from discussions with Jan-Olof Eklundh and Peter Allan.

References

1. P. Wallace, *AgeQuake: Riding the Demographic Rollercoaster Shaking BUiness, Finance and our World*. London, UK: Nicholas Brealey Publishing Ltd., 1999. ISBN 1-85788-192-3.
2. S. Thrun, "Probabilistic algorithms in robotics," *AI Magazine*, vol. 21, pp. 93-110, Winter 2000.
3. P. Jensfelt, D. Austin, and H. I. Christensen, "Towards task oriented localisation," in *Intelligent Autonomous Systems - 6* (E. Pagelle, F. Groen, T. Aria, R. Dillmann, and A. Stenz, eds.), (Venice, IT), pp. 612-619, IAS, IOS Press, July 2000.
4. M. Andersson, A. Oreback, M. Lindstrom, and H. Christensen, *Intelligent Sensor Based Robots*, vol. 1724 of *Lecture Notes in Artificial Intelligence*, ch. ISR: An Intelligent Service Robot, pp. 291-314. Heidelberg: Springer Verlag, October 1999.
5. G. Fink, N. Jungclaus, F. Kummert, H. Ritter, and G. Sagerer, "A distributed system for integrated speech and image understanding," in *Intl. Symp on Artificial Intelligence*, (Cancun, Mexico), pp. 117-126, 1996.
6. G. A. Fink, C. Schillo, F. Kummert, and G. Sagerer, "Incremental speech recognition for multi-modal interfaces," in *IEEE 24th Conf. on Industrial Electronics*, (Aachen), pp. 2012-2017, September 1998.
7. J. Cassell, "A framework for gesture generation and interpretation," in *Computer Vision for machine interaction* (R. Cipolla and A. Pentland, eds.), pp. 191-215, Cambridge University Press, 1998.
8. T. Sterner, J. Weawer, and A. Pentland, "Real-time american sign language recognition using desk and wearable computer based video," *IEEE-PAMI*, vol. 20, pp. 1371-1375, Dec. 1998.
9. R. Cipolla, P. Hadfield, and N. Hollinghurst, "Uncalibrated stereo vision with pointing for a man-machine interface," in *IAPR workshop on machine vision application*, (Tokyo), December 1994.
10. R. Cipolla, N. Hollinghurst, A. Gee, and R. Dowland, "Computer vision in interactive robotics," *Assembly Automation*, vol. 16, no. 1, 1996.
11. M. Soriano, B. Martinkauppi, S. Huovinen, and M. Laassonen, "Skin colour modelling under varying illumination conditions using skin locus for selecting training pixels," in *Real-Time Image Sequence Analysis - RISA-2000* (O. Silven and J. Heikkilä, eds.), (Oulu, Finland), pp. 43-49, Infotech, Oulu University, August 2000.
12. Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*. New York, NY.: Academic Press, 1987.
13. L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech," *IEEE Proceedings*, vol. 77, pp. 257-286, February 1989.
14. S. Young, *The HTK Book*. Cambridge University, UK, 1995.
15. L. Petersson, D. Austin, D. Kragić, and H. Christensen, "Towards an intelligent robot system," in *Proceedings of the Intelligent Autonomous Systems 6, IAS-6*, (Venice), pp. 704-709, July 2000.
16. K. Tarabanis, P. Allen, and R. Tsai, "A survey of sensor planning in computer vision," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 1, pp. 86-104, 1995.
17. S. Edelman, ed., *Representation and recognition in vision*. Cambridge, MA: The MIT Press, 1999.

18. D. Roobaert, "Improving the generalisation of Linear Support Vector Machines: an application to 3D object recognition with cluttered background," in *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI'99, Workshop on Support Vector Machines*, pp. 29–33, 1999.
19. G. Hager, W. Chang, and A. Morse, "Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination," *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 30–39, 1995.
20. W. Wilson, C. W. Hulls, and G. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 684–696, 1996.
21. E. Malis, F. Chaumette, and S. Boudet, "Positioning a coarse-calibrated camera with respect to an unknown planar object by 2 1/2D visual servoing," in *5th IFAC Symposium on Robot Control (SYROCO'97)*, vol. 2, (Nantes, France), pp. 517–523, September 1997.
22. A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 348–353, 2000.
23. K. B. Shimoga, "Robot grasp synthesis algorithms: A survey," *International Journal of Robotics Research*, vol. 15, pp. 230–266, June 1996.
24. D. Lowe, "Fitting parameterized three-dimensional models to images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 441–450, 1991.
25. D. Koller, K. Daniilidis, and H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *International Journal of Computer Vision*, vol. 10, no. 3, pp. 257–281, 1993.
26. E. Marchand, P. Bouthemy, and F. Chaumette, "A 2D–3D Model-Based Approach to Real-Time Visual Tracking," Technical report ISSN 0249-6399, ISRN INRIA/RR-3920, Unité de recherche INRIA Rennes, IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex, France, March 2000.
27. P. Wunsch and G. Hirzinger, "Real-time visual tracking of 3D objects with dynamic handling of occlusion," in *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'97*, vol. 2, pp. 2868–2873, 1997.
28. A. Miller and P. Allen, "Examples of 3D grasp quality computations," in *Proc. of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1240–1246, 1999.
29. L. Roberts, "Machine perception of three-dimensional solids," *Optical and Electrooptical Information Processing*, 1965.
30. M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Comm. ACM*, vol. 24, pp. 381–395, 1981.
31. M. Abidi and T. Chandra, "A new efficient and direct solution for pose estimation using quadrangular targets: algorithm and evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 534–538, May 1995.
32. D. DeMenthon and L. Davis, "New exact and approximate solutions of the three-point perspective problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 1100–1105, November 1992.
33. R. Horaud, B. Conio, and O. Le Boulleux, "An analytical solution for the perspective–4-point problem," *Computer Vision, Graphics and Image Processing*, vol. 47, pp. 33–44, 1989.
34. R. Tsai, "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*, vol. 3, pp. 323–344, 1987.
35. J. Yuan, "A general photogrammetric method for determining object position and orientation," *IEEE Transactions on Robotics and Automation*, vol. 5, pp. 129–142, April 1989.
36. D. DeMenthon and L. Davis, "Model-based object pose in 25 lines of code," *International Journal of Computer Vision*, vol. 15, pp. 123–141, 1995.

37. R. C. H. Araujo and C. Brown, "A fully projective formulation for Lowe's tracking algorithm," Technical report 641, The University of Rochester, CS Department, Rochester, NY, November, 1996.
38. J. Foley, A. van Dam, S. Feiner, and J. Hughes, eds., *Computer graphics - principles and practice*. Addison-Wesley Publishing Company, 1990.
39. S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 651–670, 1996.