A Framework for Visual Servoing Tasks

D. Kragić[1] and H. I. Christensen[2]
[1] Computer Vision and Active Perception [2] Centre for Autonomous Systems
Numerical Analysis and Computing Science
Royal Institute of Technology, Stockholm, Sweden
danik,hic@nada.kth.se

Abstract. A general framework for visual servoing tasks is proposed. The objective of the paper is twofold: a) how a complicated servoing task might be composed from a multitude of simple ones, and b) how the integration of basic and simple visual algorithms can be used in order to provide a robust input estimate to a control loop for a mobile platform or a robot manipulator.

For that purpose, voting schema and consensus theory approaches are investigated together with some initial vision based algorithms. Voting is known as a model–free approach to integration and therefore interesting for applications in real–world environments which are difficult to model. It is experimentally shown how servoing tasks like pick–and–place, opening doors and fetching mail can be robustly performed using the proposed approach.

1. Introduction

In the field of service robotics, robots should continuously interact with objects and human beings in a natural, unstructured and dynamic environment. In a general case, it might not be possible or feasible to know *a-priori* the state of the outside world. For such systems, machine vision is an important sensory modality. The applications range from tasks like cleaning to preparing meals and helping disabled persons in their everyday life.

The ability to perform manipulation tasks is a key issue where the flexibility and robustness are both the primary goals and the obstacles. Tasks like fetch and carry, opening doors, pick and place are some of the examples we want our robots to perform. These tasks motivate the work presented in this paper.

During the past few years, hardware has gradually reached a level of sophistication that allows real–time implementations of vision based algorithms which were in the past usually considered as slow, computationally heavy and for that reason were usually tailored to perform dedicated tasks or even abandoned. Notable exceptions include Dickmanns [7], and Nagel et al [9]. Both of these approaches have adopted specific models (in terms of the environment and/or the objects of interest (cars)). In terms of manipulation most approaches exploit markers to simplify detection and tracking. Examples of such work include Hager [8], Rives [4, 16] and Allen [1].

The motivation for our work have been the following questions: If a single visual process can increase the autonomy of a system, would the use of a number of simultaneous processes increase it even further? and Is it possible to overcome the weaknesses due to dependencies and constraints of individual processes?

Over the past decade a number of researchers have been exploring these questions from both a theoretical and an implementation perspective. We reference some of this work in Section 2. In Section 3. a voting schema approach is presented together with the chosen visual cues. Section 4. presents the experimental setup and experimental results and, finally, the conclusion and ideas for future work are given in Section 5.

2. Related Work

Clark and Yullie [6] classified methods for integration of different vision modules into *weak coupling* and *strong coupling*. Weak coupling combines the outputs of different modules while in strong coupling the output of one module affects the output of another module. Which integration method to use, depend mainly on the modules to be integrated. In some cases, to improve the reliability of a system, a weighted combination of modules might be sufficient. Methods for combination of independent or dependent modules have been studied extensively in the literature.

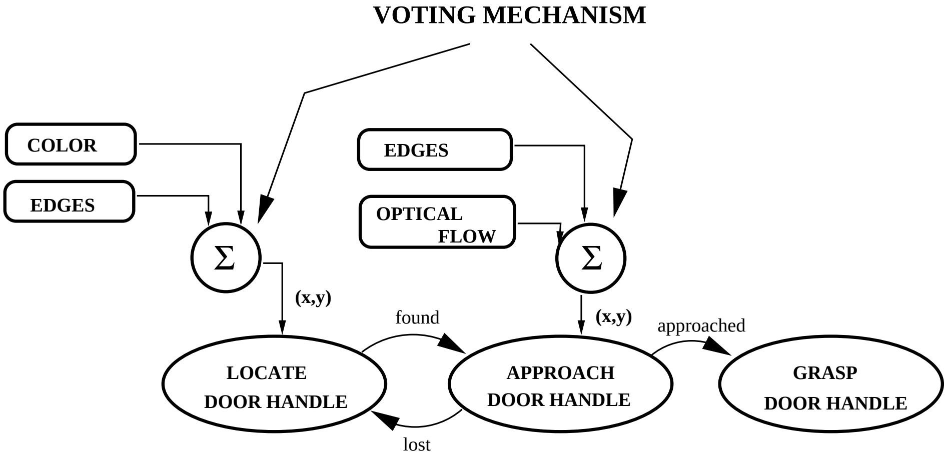


Figure 1: An example of a state machine for locating and grasping of a door handle.

Probabilistic methods such as Bayesian approach [6], Kalman filters [20], Dempster–Shafer theory [12] have been widely used in fusion of multiple cues. In computer vision this approach was mainly used for scene reconstruction and pattern recognition. Probabilistic methods are robust and have moderate complexity in cases where prior distributions of possible results are available. However, it is very difficult to determine an accurate and correct model, e.g. goodness–of–fit of the model to the data is a crucial prerequisite to achieve robustness and low complexity.

Rule-based integration methods successively generate more specific interpretations of data based on the strength of the output of analysis modules, *a–priori* constraints and further expectations. The generation is led by rules which encode the uncertainty of the underlying data and the ambiguity of interpretations. The interpretations with highest confidence influence the choice of the rule to be applied. Pridmore et al. [17] used a rule–based system for 3D grouping of disparities. Nazif and Levine [13] designed a rule–based system for solving the image segmentation problem. A major drawback of rule–based systems is that rules must be generated in advance. Similarly to probabilistic approach, an accurate and correct model of the environment is needed.

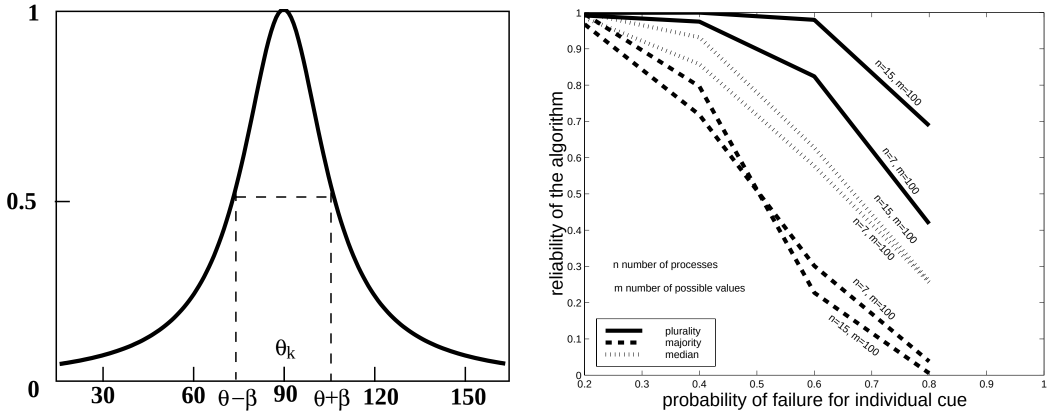
When exact models of the individual input data, their error rates and confidence measures are not available, voting can be used to improve the reliability of the overall system, [14, 15]. Voting is commonly used in systems where fault tolerance is required. In voting applications, the processes in a system generate values that are communicated to a voter. Given the data, the voter must output a single value from the set of possible values. A recent trend in voter–design is to use probabilistic information to improve voting performance, [5]. Weighted voters are an attempt to account for distinct failure probabilities, [21]. Although there are many proposed variations of voters, the most commonly used ones are median and majority voters. Extensive work has been done on how to evaluate and improve these basic algorithms, [2].

Blough and Sullivan, [5], distinguish between *exact* (a single value is correct) and *inexact* voting situations (multiple values exist). In their paper, they present a comprehensive probabilistic model for the voting problem. Their model incorporates four distinct probability distributions which can represent the behavior of faulty and non–faulty modules as well as any underlying probability distribution on the space of answers. They compare the optimality of several commonly used voting algorithms and reveal the desirable characteristics of the plurality voting schemas.

Our strategy uses the voting methodology to achieve agreement between a number of visual processes. For that purpose, we define a common voting space that is equivalent to the image space. This implies that the different visual processes vote for a particular characteristic of each pixel in the image space and provide a vote for each class (i.e., determine if a pixel belongs to a vertical edge, if the color of a pixel is red, if the pixel is part of an moving region, etc.). These votes might be binary or distributed around the desired value. In the following section, we present some implementation details.

3. Implementation

A typical task like opening doors and picking up a cup from a table can be structured as a multitude of sub-tasks, see Fig. 1.



(a) An example of a Π-function when θ is gradient direction with the desired value $\theta_d = \pi/2$ and allowed deviation $\beta = \pi/9$

(b) reliability comparison of Majority, Median and Plurality Voting

Figure 2:

As already mentioned in the previous section, our idea is to use basic/simple vision algorithms that are most probable to give a response during the current task. Some of the tasks might be: figure–ground segmentation (color, shape, inverse–perspective mapping), tracking (optical–flow, image differencing, color, disparity), grasping (edges, corners), etc.

In our previous work, [10], we described and tested how information from multiple visual sources can be combined into a robust visual servoing system for tracking. This task usually arises when there is a moving camera and the goal is to focus the attention to a particular region in the image or while performing an approach maneuver in a grasping task. Both of these examples require continuous update of the position of the region of interest in the image. Using *a–priori* information about the object (color, shape, velocity), a position estimate can be performed using responses of different visual cues which are integrated into a common voting space.

3.1. Integration Approach

Notation:

P	set of processes that produce results to be voted on
n	number of processes in the system
V	set of possible values that can be produced by a process
X	action space (image space, joint space)
S	S=(P, V, X) a system
i_k	process k
θ_k	process–specific information class (color, edge,motion)
Z	$Z = [z_1, ...z_n]$ vector of observations from all data sources
p	$p(\theta_k \| z_k)$ process–specific posterior probability
λ_k	process weights
$C(Z)$	global membership function

A system S is comprised of a set of processes P. Each process, i_k, produces a set of possible values, V. The processes are either homogeneous (the probability of a failure is equal for each process) or heterogeneous (the individual failure probabilities are different). The probabilities are expressed through weight factors λ_k and

$$\sum_{i=1}^{n} \lambda_k = 1 \tag{1}$$

A process i_k is a mapping from an action space, X, to the set of possible values V. Set V can be a binary set where the desired values are assigned 1 and undesired values are assigned 0:

$$i_k : X \to V \quad where \quad V = \{0, 1\} \tag{2}$$

An alternative to binary mapping might be to formulate the mapping using a Π-function:

$$\Pi(\theta) = \frac{1}{1 + (\frac{\theta - \theta_d}{\beta})^2} \qquad where \qquad V = [0, 1] \tag{3}$$

where θ_d is the desired value and β is allowed deviation, see Fig. 2(a).

There are several voting methods commonly used in the literature. Blough and Sullivan, [5], presented a reliability comparison between majority, median and plurality voting, see Fig. 2(b). The majority voting algorithm chooses a value θ_k as correct iff it is produced by more than half of the processes. The median voting algorithm chooses the median value from a collection of results. The plurality voter picks the value that occurs most frequently. These algorithms are well suited for systems where the number of processes $n \geq 3$.

In our approach, there is usually a case of two processes and set of possible values being either $V = \{0, 1\}$ or $V = [0, 1]$. We use some ideas from consensus theory to develop voting rules, [3]. In consensus theory, the information from different processes is aggregated by a global membership function and the data are classified according to the usual maximum selection rule into the information classes. The combination formula obtained is called a consensus rule. The two most commonly used consensus rules are:

$$\textbf{Linear opinion pool (LOP):} \qquad C(Z) = \sum_{k=1}^{n} \lambda_k p(\theta_k | z_k) \tag{4}$$

$$\textbf{Logaritmic opinion pool (LOGP):} \qquad C(Z) = \prod_{k=1}^{n} p(\theta_k | z_k)^{\lambda_k} \tag{5}$$

After the aggregation using a global membership function, the data are usually classified using the maximum selection rule.

In image space, LOP and LOPG might be used in two different ways:

1. For each pixel, I(x,y), posterior probabilities $p(\theta_k | z_k)$ are computed. A consensus rule is then used individually for each pixel giving a probability value for a pixel of simultaneously belonging to the desired classes.

   This approach is suited for implementations where a 2D probability distribution in voting space is needed. An example might be where we are given a 2D model of a target and we want to fit it to the image data.

2. For each process i_k, a consensus rule is used to find one pixel in the image plane with the highest probability of containing the desired value. After that, a consensus rule might be used again to integrate the outputs from individual processes.

   An example of this approach might be as follows: assume a few different tracking algorithms based on optical flow, color, edges, etc. Each algorithm outputs the most probable position of the tracked region:

$$\theta'_{x_k} = \frac{\sum_{xy} x p(\theta_k | z_k)}{\sum p(\theta_k | z_k)}, \qquad \theta'_{y_k} = \frac{\sum_{xy} y p(\theta_k | z_k)}{\sum p(\theta_k | z_k)} \tag{6}$$

   A triangular function $f_T(\theta'_k)$ with the peak at the image position $(\theta'_{x_k}, \theta'_{y_k})$ is then developed as the output of each algorithm. Finally, a consensus rule is used to integrate these outputs:

$$C(Z) = \sum_{k=1}^{n} \lambda_k f_T(\theta'_k) \tag{7}$$

4. Experimental Evaluation

4.1. Experimental Setup

The experiments were conducted with a Puma560 robotic arm in two different settings: a XR4000 mobile platform with a Puma560 on top, Fig. 3(b) and a “static” Puma560 attached to a table, Fig. 3(a).

The camera settings were as follows: in the case of the mobile platform the camera is attached on the gripper (eye-in-hand configuration) while in the second case we are using so called ”stand-alone” system where the cameras are attached on the side of the table and monitor both the manipulator and the workspace at the same time.

4.2. Opening Doors

This particular experimental sequence starts when a robot platform is positioned approximately in front of a door to be open. The door handle is modeled in two different ways: as a rectangular region of mostly uniform color and a cross–like set of lines, see Fig. 4. This task comprises three different states as presented in Fig. 1.

During *locate and approach state* there are two parallel processes: color segmentation and gradient computation, ∇g_x. During the first few frames, a color histogram is obtained. With the assumption that the camera faces a door handle, a color histogram will contain two distinct peaks: one for the door and one for the door handle color. We choose a grey level value (GLV) of the darker peak to be desired GLV, θ_1 for a color segmentation process, i_1. Posterior probabilities, $p(\theta_1|z_1)$ are computed using a Π-function distributed around the desired GLV. For the second process, i_2, the desired value, θ_2, will be the gradient direction $\pi/2$. Processes are assumed to be failure free, $p(z_i)$=1 and to have equal reliability, $\lambda_1 = \lambda_2$ =0.5. For each pixel in the image, LOP is used to find a probability that a pixel belongs to a vertical edge and contains the right GLV. Using a door handle model, a region that fits best to the model is chosen.

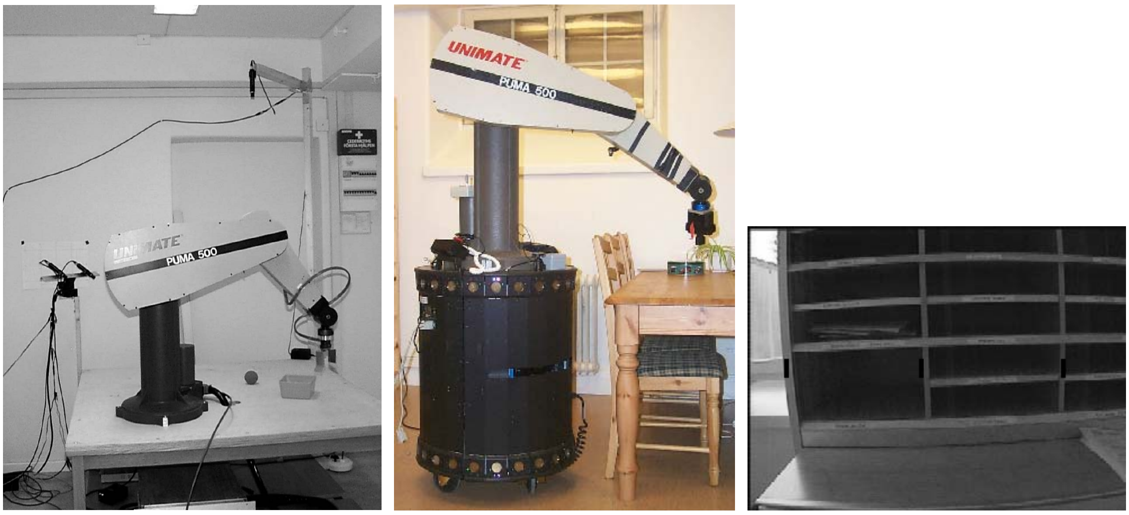
When the door handle is detected, the state is changed to the *approach door handle* state. A template around the door handle is initialized and used for template based optical flow tracking. The output of the tracker is the image position (x,y) of the door handle. The second process, gradient computation ∇g_x, is used to detect center of mass of horizontal edges. Both processes output an image position which is used as a peak of a triangular function. Again, LOP is used as a voting mechanism to integrate the inputs of these two processes.

As a stopping criterion, we use the width between the two strongest vertical edges. When its size reaches a predefined threshold, the *grasp door handle* state can start. During this state, a force-torque sensor is used to perform the grasping task.

4.3. Fetching Mail

Similar to the previous example, we chose to divide this task into the following states:

1. locate
2. approach – use vertical lines
3. approach – use name tag
4. fetch



(a) Puma560 arm attached to a table

(b) Nomad XR4000 with a Puma560 on the top

(c) A view on the ”mail cupboard” where a persons mailbox is positioned in the left, middle or right part of the cupboard

Figure 3:

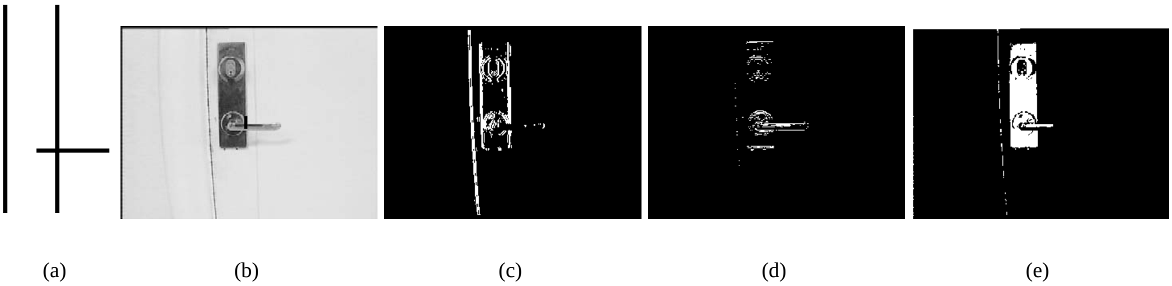


Figure 4: a) a door handle model, b) a typical view on a door handle, c) ∇g_x, d) ∇g_y, e) color segmentation

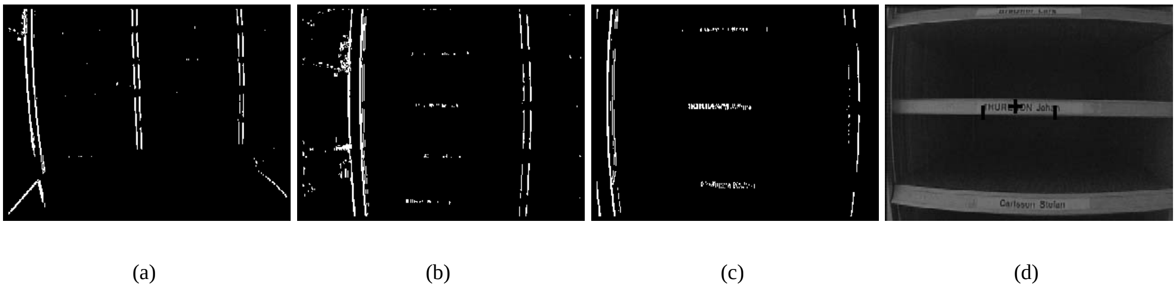


Figure 5: a) ∇g_x when the whole "mail cupboard" is visible, b) ∇g_x when the robot started to approach the right mailbox, c) ∇g_y when the robot is at about 30cm from the mailbox, d) the name tag is used to guide the robot forward

This task starts when the platform is already in front of the “mail cupboard” and the relative position is known with a certain accuracy. The mail cupboard consists of a number of mailboxes and each mailbox contains a name tag at the bottom, see Fig. 3(c). We made a look–up table where with each name there is a position of the persons mailbox (left, middle or right) with some initial arm position. The reason for doing that is that at this stage we do not perform search for the name on the mailbox.

The mail cupboard is a typical case of a repetitive pattern. Vertical and horizontal lines connect the regions of mostly uniform color (mailboxes). During the *locate* state, we use the gradient information ∇g_x in order to select the strongest vertical lines. The hypothesis for the whole cupboard is made where the regions of uniform color are found between the strongest vertical lines. Depending on the position of the strongest vertical lines, we are able to position the platform in front of the right part of the mail cupboard (left, right or middle), see Fig. 5. Once the mail cupboard is detected, the platform is moved left or right depending on which side the persons mailbox is situated at. This state is finished when the platform is centered in front of the mail cupboard.

The automata switches next to the *approach* state. During this state the platform moves forward while centering the mailbox in the image. During this stage, in each pixel ∇g_x is computed. The desired GLV in this case is chosen *a–priori*. The integration is done in the same way as in the case of *locate and approach state* in the door opening example.

When the platform is at about 30cm from the mail box, the vertical lines fall outside the field of view of the camera and the third state is initiated. During this state we concentrate on the name tag since this is almost the only textured part of the image. The initial position of the arm is such that the name tag is approximately in the middle of the image so the search for a name tag does not have to be performed over the whole image. Since we do not use character recognition at this stage, we compute ∇g_x and ∇g_y in each pixel around the image center. The center of the name tag can be easily found using Eq. 6 and Eq. 7.

When the size of the name tag reaches the predefined size in the image, the state it changed to the *fetch* state. Again, a force–torque sensor is used to perform the grasp maneuver. Some of the images taken while performing the task are presented in Fig.5.

4.4. Manipulating Objects

Some of the ideas presented in the two previous sections, have also been implemented for simple pick–and–place tasks, see Fig 6. In both cases a stand–by camera system is used as explained in Section 4.1.

A typical pick–and–place task consists of the following states:

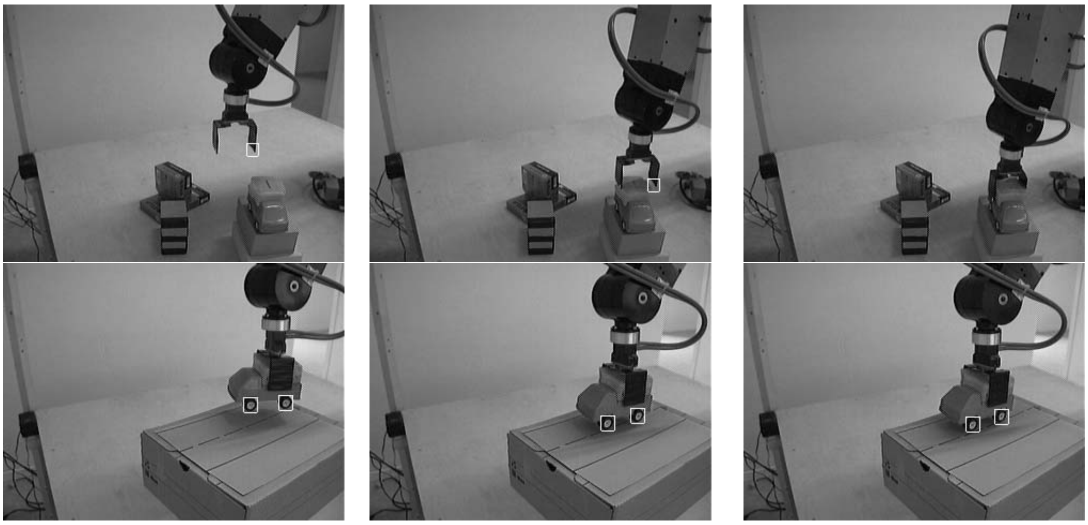


Figure 6: Grasping and positioning a car parallel to the road.

1. locate the object on the table:
   The simplest approach uses color and shape to find a region in the image that is most likely the object of interest. As the position of the object the center of mass might is used. Second order moments are used to retreive the orientation.

2. pick–up the object:
   After determining the pose of the object, two pairs of trackers are initiated: one on the object and one on the gripper. The distance between the trackers in image space is used as input to a control loop.

3. place the object:
   The position of the object together with the desired final position are used as the input to the control loop.

5. Conclusions

A general framework for integration of basic visual tasks was presented. The objective for the research was twofold: i) to determine a framework for division of complex visual tasks into a number of atomic tasks that exploit simple visual servoing methods for control, and ii) to explore methods for fusion of the atomic tasks to increase the robustness of the complete task.

Earlier reported research has clearly demonstrated how integration of multiple cues facilitate robust tracking of objects in cluttered scenes [19, 15]. The focus of this paper has thus been on formulation of an automata approach to task decomposition and the combination of this framework with robust cue integration methods that allow tracking in situations where each of the atomic tasks would fail if applied individually. It has been demonstrated how consensus theory can be used to develop a methodolody for fusion of tasks. In parallel, the automata model provides a suitable model for macroscopic control of the servoing tasks. The utility of the presented approach was illustrated by the three tasks: of door-opening, mail-pickup and object manipulation. For all three cases a task decomposition was presented together with the used methods for fusion of visual cues for the control.

The presented methodology for task decomposition leans itself to analysis in terms of liveliness, deadlocks, reachability using methods from discrete event control [18], while the fusion framework enables use of diagnostic methods for determination of failure of individual and collections of visual processes, which can be based on traditional decision theory [11]. Future work will in particular explore use of such theoretical tools for design and analysis of visual servoing tasks.

Acknowledgment

This research has been sponsored by the Swedish Foundation for Strategic Research through its Centre for Autonomous Systems. The funding is gratefully acknowledged. The research reported on mobile manipulation has benefitted from control software developed by Lars Petersson, Patric Jensfelt and David Austin.

References

[1] P. Allen. Automated tracking and grasping of a moving object with a robotic hand-eye system. *IEEE TRA*, 9:152, 1993.

[2] D. Barbara and H. Garcia-Molina. The reliability of voting mechanisms. *IEEE Transactions on Computers*, C-36:1197–1208, October 1987.

[3] J.A. Benediktsson, J.R. Sveinsson, and P.H. Swain. Hybrid consensus theoretic classification. *IEEE Transactions on geoscience and remote sensing*, 35(4):833–842, 1997.

[4] B.Espiau, F.Chaumette, and P.Rives. A new approach to visual servoing in robotics. *IEEE TRA*, 8(3):313–326, 1992.

[5] D.M. Blough and G.F. Sullivan. Voting using predispositions. *IEEE Transactions on reliability*, 43(4):604–616, 1994.

[6] J. Clark and A. Yuille. *Data fusion for sensory information precessing systems*. Kluwer Academic Publisher, 1990.

[7] Ernest Dickmanns. Vehicles capable of dynamic vision: a new breed of technical beings? *Artificial Intelligence*, 103(1–2):49–76, August 1998.

[8] G.D. Hager. Calibration-free visual control using projective invariance. *Proc. ICCV*, pages 1009–1015, 1995.

[9] H. Kollnig and H.H. Nagel. 3d pose estimation by directly matching polyhedral models to gray value gradients. *International Journal of Computer Vision*, 23(3):282–302, 1997.

[10] D. Kragić and H.I Christensen. Active visual tracking of an end-effector: Integration of various cues. *Proc. IEEE IROS*, 1:362–368, 1999.

[11] Steen Kristensen and Henrik I. Christensen. Decision-theoretic multisensor planning and integration for mobile robot navigation. In Masatoshi Ishikawa, editor, *Proceedings of the 1996 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI'96)*. IEEE Computer Society, December 1996.

[12] C. Lee. A comparison of two evidental reasoning schemas. *Artificial Intelligence*, 35(1):127–134, 1988.

[13] A.M. Nazif and M.D. Levine. Low level image segmentation: an expert system. *IEEE Trans. PAMI*, 6(5):555–577, 1984.

[14] B. Parhami. Voting algorithms. *IEEE Transactions on Reliability*, 43(3):617–629, 1994.

[15] P. Pirjanian, H. I. Christensen, and J.A. Fayman. Application of voting to fusion of purposive modules: an experimental investigation. *Robotics and Autonomous Systems*, 23(4):253–266, 1998.

[16] R. Pissard-Gibollet and P. Rives. Applying visual servoing techniques to control a mobile hand-eye system. In *Proceedings IEEE Robotics and Automation, Nagoya, Japan*, pages 725–732, October 1995.

[17] T.P. Pridmore, J.E. Mayhew, and J.P. Frisby. Production rules for grouping edge based disparity data. *Alvey Image Understanding Conference*, 1985.

[18] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–97, January 1989.

[19] C. Rasmussen and G. D. Hager. Joint probabilistic techniques for tracking objects using multiple visual cues. *Proc. IEEE/RSJ IROS*, 1(191-196), 1998.

[20] D. Terzopoulos and R. Szeliski. Tracking with Kalman snakes. In A. Blake and A. Yullie, editors, *Active Vision*, pages 3–20. MIT Press, Cambridge, MA, 1992.

[21] Z. Tong and R. Kain. Vote assigments in weighted voting mechanisms. *IEEE Transactions on Computers*, 40:664–667, 1991.