# Door Opening: A Case Study in Robotic Systems Integration

Lars Petersson, David Austin, Danica Kragić

*Keywords*— **Systems integration, mobile manipulation, visual servoing, door opening**

*Abstract*— **In this paper, we study the integration of a variety of different disciplines to produce a robotic system with the ability to perform a complex, real-world task. The task to be studied is to open a door. For a robust solution to this task, we must use computer vision to locate the door handle and move the robot close enough to grasp the handle. Once a grasp has been established, we must combine the motions of the manipulator with the motion of the mobile base to successfully open the door. In this paper, simple algorithms for force/torque control are used in the context of mobile manipulation (i.e. coordinated motion of a mobile robot base and the arm mounted on top of it). To make the solution robust, as few assumptions as possible are made. By using relaxation of forces as the basic level of control, more complex information can be derived from the resulting motion. In our system, the radius and the centre of rotation of the door are estimated online. The integration of the above approaches results in a system which can robustly locate and grasp the handle and then open the door.**

## I. Introduction

FORCE control and compliant motion for very simple tasks in industry are well studied areas [1]. However, the higher levels of control required for complex, real-world tasks need further study. In this paper, we discuss how a number of basic sensor and manipulation methods can be integrated to perform complex tasks. In the field of mobile robotics, manipulation skills are often necessary to perform useful tasks, e.g. fetch and carry, open doors and drawers etc. Manipulation using a *mobile* manipulator introduces a new set of problems that are not well understood. Simultaneous control of both the mobile base and the manipulator arm has only been studied by a few researchers (e.g. [2], [3]). The objective of this paper is to go to the next level, i.e. by combining relatively simple force control and vision algorithms, what can be achieved through proper integration? A typical, useful task for a mobile manipulator is to open a closed door so that it can pass through. Therefore, this particular task was chosen to be investigated in this paper.

In robotics, there is a tendency to construct specific solutions (hacks) for every task that occurs. For example, previous door opening approaches [4] assume that the position and radius of the door are well known. Another door opening system has been presented, called ROMAN [3], but almost no information is provided as to how ROMAN opens the door. In the spirit of making a general purpose door opening algorithm, we assume that we only know the approximate position of the door (given by a localisation module, for example) and use a visual servoing algorithm to establish a grasp on the handle. We also make no assumptions concerning the radius of the door or the position of the hinges. Instead, these parameters are estimated from the motion of the robot when a compliant controller is used.

L. Petersson, D. Austin and D. Kragić are all with the Center for Autonomous Systems, Royal Institute of Technology, Stockholm, Sweden. E-mail larsp, daustin, danik@nada.kth.se

The manipulator arm and the mobile platform is controlled through the use of a hybrid dynamic system. Although not used in here, this opens up for the possibility to make a formal, theoretical study of the problem within the framework of hybrid dynamic systems. The work presented here uses a package developed earlier at the Centre [5] to implement the control strategies.

The next section presents the problem, describing the class of doors we wish to open and the hardware of the robot available to us. In Section III, the general approach used and a brief introduction to hybrid dynamic systems is given, followed by a description of the actual hybrid dynamic system that is used for door opening. Section IV describes the visual servoing algorithm which is used to establish a grasp on the door handle. Section V presents the manipulation strategies used as well as the door parameter estimation technique and the method used for managing the redundant degrees of freedom of the robot. Section V-C presents the experimental results from the parameter estimation and, finally, Section VII discusses the results and compares our system with other approaches.

## II. Problem

The problem that we decided to solve is to open a door so that the robot may pass through it. This problem requires similar manipulation to other tasks for a mobile robot and so is a good test case to study. Figure 1 shows a picture of the door handle. These handles are used throughout our building. This style of handle poses problems for the vision algorithms because the metallic surfaces give good reflections of the robot itself. The robot is a Nomadic Technologies XR4000 and is equipped with a Puma 560 arm for manipulation (see Fig. 2). The robot has two rings of sonars, a SICK laser scanner, a wrist mounted force/torque sensor (JR3), fiber optic gyros with accelerometers (DMU-FOG) and a color CCD camera mounted on the gripper. Note that, due to the camera placement, it is not possible to keep the door handle in camera view when gripping it.

The system is controlled by three different on-board computers (two 550MHz Pentium III and one 233MHz Pentium), each one dedicated to a specific task. The Puma 560 is controlled from a computer which is running the

Fig. 1. Close-up of the door handle.


Fig. 2. The hardware on which the algorithm was implemented.

RTOS QNX due to the real-time considerations that are important in a system with fast dynamics. This computer also runs the complete hybrid automata described later on. The XR4000 base is controlled from a Linux computer using the standard libraries provided by Nomadics. The third computer which is also running Linux is dedicated to computationally heavy duties such as computer vision.

### A. Software

The mobile base is running an integrated system that incorporates higher level functionalities such as localization, navigation and world modeling. These parts of the system are described in detail in [6]. All software is written in C++ and uses the client-server paradigm throughout the system. This gives modularity and the possibility to easily change different parts.

Manipulation skills comes in as a natural extension to the system which is presented in [5]. This extension consists of a framework to write hybrid automata for a unified control of the platform and manipulator.

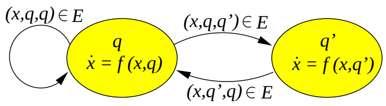



Fig. 3. A simple hybrid dynamic system.

## III. Approach

To solve this problem, we decided to combine a number of simpler operations into an integrated system using the hybrid dynamic system framework. A hybrid dynamic system [7] consists of a mixture of continuous and discrete parts. Formally, a hybrid automaton is considered to be a collection $(Q, X, I, f, E)$ where $Q$ and $X$ are sets of discrete and continuous variables respectively. $I$ is a set of initial conditions, $f$ describe the continuous and $E$ the discrete evolution of the states.

Figure 3 shows a simple example of a hybrid dynamic system. The basic idea is that a hybrid dynamic system consists of sets of continuous dynamics with switches between them caused by unpredictable and sudden *events*. Note that the hybrid dynamic system framework allows the combination of different control strategies by sequencing or switching between them. In this way, the hybrid dynamic systems framework permits modular controller design. In addition, it is possible to answer theoretical questions such as safety for hybrid dynamic systems. Other models for combining controllers, such as behaviour-based schemas do not have this possibility.

Figure 4 shows the hybrid dynamic system designed for the door opening task. The problem of opening the door was decomposed into the following steps: (i) move to a position in front of the door, (ii) find the door handle using the CCD camera, (iii) and servo on the door handle, (iv) grasp the handle, (v) rotate the handle, and (vi) open the door, as shown in Figure 4. Step (i) is accomplished using the general localization and navigation modules of the ISR system [6] and step (ii) uses the algorithm described in [8] which matches a number of features to the door handle to find the door handle in the image. The robot then servos towards the door handle (Step (iii)) [8], [9]. Since vision does not provide very accurate depth information, the laser range sensor is used to measure the distance to the door. When the end-effector is about 8cm away from the handle, it is necessary to do a blind grasp since the handle is no longer completely visible in the image. This grasp continues in the depth direction until the force sensor indicates that contact is made (step (iv)). The gripper is then closed around the door handle, and the door handle is rotated and then the actual door opening algorithm starts.

### A. Opening the Door

At the time of pulling the door open, no assumptions are made other than that the previous steps have successfully placed the end-effector close to perpendicular to the door handle and that a firm grip has been established. A straight line motion is then initiated, i.e. along an arc with infinite radius. Due to the slightly incorrect motion, this will exert forces and torques at the end-effector which have

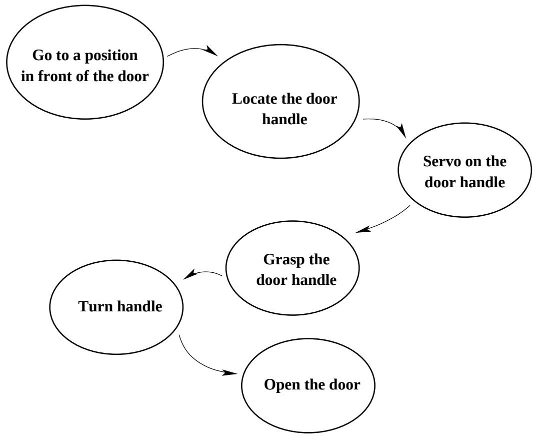



Fig. 4. A diagram of the hybrid dynamic system used for the door opening.

to be relaxed in order to continue the task. Relaxing the contact forces and torques implies changing the direction of the end-effector in such a way as to follow the correct arc. The radius and the centre of rotation will be further estimated from the recorded motion as the door opening continues. The newly estimated radius will then be used in a predictive controller to improve robustness and speed. To avoid stretching of the manipulator arm, the system also tries to achieve a desired manipulator configuration. This algorithm is presented in detail in Section V.

## IV. Visual Servoing

As already mentioned, a door opening task can be structured as a number of sub-tasks, see Fig. 4. The basic idea of our approach is to use simple vision algorithms that are most likely to give a response during the current subtask. Some of the subtasks may 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 experimentally tested how multiple visual algorithms/cues may be combined into a robust visual tracking system for servo control of a manipulator. Using *a–priori* knowledge about the object (color, shape, velocity), a continuous update of the object's position in the image is estimated based on responses of different visual cues that are integrated into a common voting space.

In this work, voting methodology is again used for achieving the agreement between a number of concurrent visual algorithms/cues. For that purpose, we define a common voting space that is equivalent to the image space. This implies that different visual algorithms vote for a particular characteristic of each pixel in the image 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 a pixel is part of a moving region, etc.). These votes may be binary (yes/no answer) or distributed around the desired value.

### A. Background and Related Work

Clark and Yullie [11] 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. Integration methods have been widely studied in the literature. Probabilistic methods such as Bayesian approach [11], Kalman filters [12], Dempster–Shafer theory [13] 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 methods, [14],[15], 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 data and the ambiguity of interpretations. A major drawback of rule–based systems is that rules must be generated in advance.

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, [16], [17]. 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. Weighted voters are an attempt to account for distinct failure probabilities, [18],[19]. Blough and Sullivan, [18], distinguish between *exact* (a single value is correct) and *inexact* voting situations (multiple correct values exist). They present a comprehensive probabilistic model for the voting problem. They compare the optimality of several commonly used voting algorithms and reveal the desirable characteristics of the plurality voting schemas.

### B. Implementation

**Notation**:

| | |
|---|---|
| $P$ | set of processes (algorithms) 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 |
| $\theta_k$ | process–specific information class (color, edge,motion) |
| $Z$ | $Z = [z_1, ...z_n]$ vector of observations from all data sources |
| $p$ | $p(\theta_k \mid z_k)$ process–specific posterior probability |
| $\lambda_k$ | process weights |
| $C(Z)$ | global membership function |

A system $S$ is comprised of a set of processes (algorithms), $P$. 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 $\lambda_k$ and $\sum_{i=1}^{n} \lambda_k = 1$. 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:

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

An alternative to binary mapping may be to formulate the mapping using a $\Pi$-function:

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

where $\theta_d$ is the desired value and $\beta$ is allowed deviation, see Fig. 5(a). There are several voting methods commonly used in the literature. Blough and Sullivan, [18], presented a reliability comparison between majority, median and plurality voting, see Fig. 5(b). The majority voting algorithm chooses a value $\theta_k$ 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 the set of possible values being either $V = \{0, 1\}$ or $V = [0, 1]$. We use some ideas from consensus theory to develop voting rules, [20]. 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 commonly used consensus rules are *Linear opinion pool* **(LOP)** and *Logarithmic opinion pool* **(LOGP)**:

$$\textbf{LOP}: \qquad C(Z) = \sum_{k=1}^{n} \lambda_k p(\theta_k | z_k) \tag{3}$$

$$\textbf{LOGP}: \qquad C(Z) = \prod_{k=1}^{n} p(\theta_k | z_k)^{\lambda_k} \tag{4}$$

After the aggregation using a global membership function, the data are usually classified using the maximum selection rule. In image space, LOP and LOGP may 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 two-dimensional probability distribution in voting space is needed. An example is a case where we are given a two-dimensional model of the 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 with the highest probability of containing the desired value. After that, a consensus rule may be used again to integrate the outputs from individual processes.

An example of this approach may 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:

$$\theta'_{x_k} = \frac{\sum_{xy} x p(\theta_k|z_k)}{\sum p(\theta_k|z_k)}, \qquad \theta'_{y_k} = \frac{\sum_{xy} y p(\theta_k|z_k)}{\sum p(\theta_k|z_k)} \tag{5}$$

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:

$$C(Z) = \sum_{k=1}^{n} \lambda_k f_T(\theta'_k) \tag{6}$$

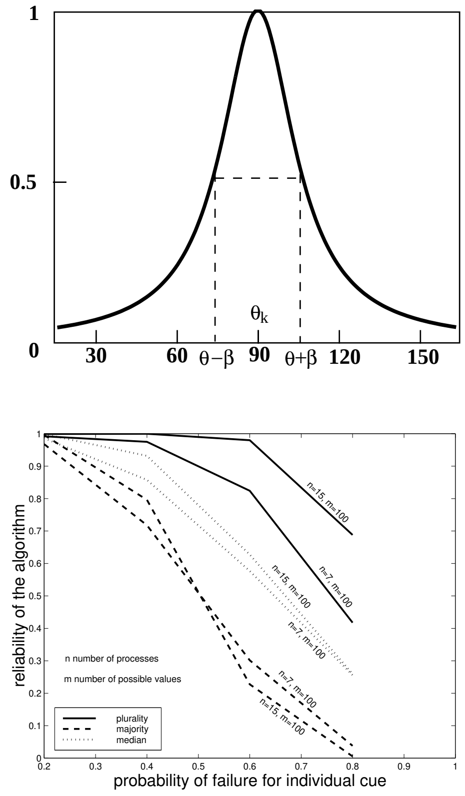



Fig. 5. Upper: An example of a $\Pi$-function when $\theta$ is gradient direction with the desired value $\theta_d = \pi/2$ and allowed deviation $\beta = \pi/9$, Lower: Reliability comparison of Majority, Median and Plurality Voting

### C. Experimental Results

This particular experimental sequence starts when a robot platform is positioned approximately in front of the door to be opened. 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. 7.

During *Locate the door handle state* there are two parallel processes: color segmentation and gradient computation, $\nabla g_x$. During the first few frames, a color histogram

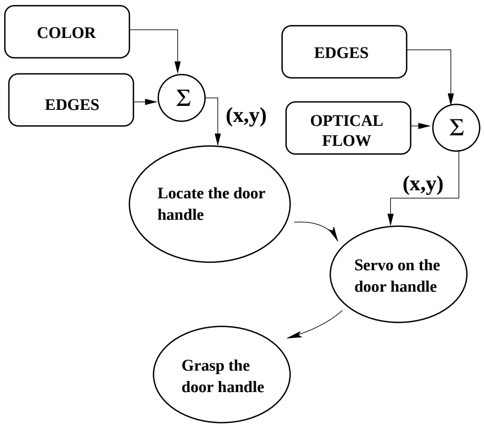



Fig. 6. Voting mechanisms used in the hybrid dynamic system for door opening.

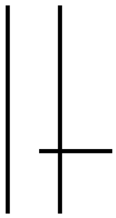


Fig. 7. The door handle model used

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, $\theta_1$ for a color segmentation process, $i_1$. Posterior probabilities, $p(\theta_1|z_1)$ are computed using a $\Pi$-function distributed around the desired GLV. For the second process, $i_2$, the desired value, $\theta_2$, 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 *Servo on the 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, the gradient computation $\nabla\, g_x$, is used to detect the 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.

Fig. 8 shows an example of the images from the camera and the processing steps discussed above. There are two major problems that we have to cope with: the properties of the material of the door handle and the simplicity of the handle model used. The metal of the door handle is highly reflective which significantly affects the color segmentation part. Simple histogramming can be used to determine two peaks in the color histogram which would imply the distinction between the door handle (darker) and the door itself (lighter). However, if the lighting is too bright, there are no distinct maximas in the color histogram. Because of the simplicity of the handle model, it is very difficult to detect false matches. Extensive experiments show that these two problems result in an offset of the end–effector from the desired position relative to the door handle. However, this offset is usually less than 1cm, which is small enough to successfully perform the task.

## V. Manipulation

The notation in the following sections is as follows; $F^{Base}$ and $F^{EE}$ means forces and torques with respect to a coordinate system fixed at the base and the end-effector respectively. $T$ denotes general coordinate transformations while $R$ denotes pure rotations. $T^{Room}$ would denote a transformation that relates points to a fixed coordinate system in the room. Subscripts are used to denote individual components, e.g. $F_x$ is the force along the x-axis.

### A. Door Modeling

The basic type of motion is assumed to be planar and following an arc. Therefore the model of the door lies in the x-y plane of the base and consists of a centre of rotation and a radius (Figure 9).

This means that the following relation must hold at all times

$$r^2 = (x - x_0)^2 + (y - y_0)^2 \tag{7}$$

where $(x_0, y_0)$ is the centre of rotation and $r$ is the radius with respect to the point of contact with the gripper $(x, y)$.

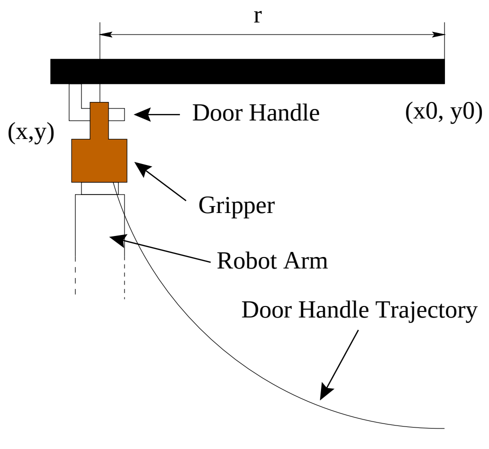



Fig. 9. The planar model of the door consists of the centre of rotation and radius.

#### A.1 On-line Parameter Estimation

The idea is to estimate the parameters of the model above by using the recorded motion of the end-effector.

Equation 7 is rewritten to look like

$$x^2 + y^2 = r^2 - {x_0}^2 - {y_0}^2 + 2xx_0 + 2yy_0 \tag{8}$$

Fig. 8. From left to right: a typical view of a door handle, $\nabla g_x$, $\nabla g_y$ and color segmentation

The matrix form becomes

$$At = b \tag{9}$$

where $A = \begin{pmatrix} 1 & x & y \\ & \vdots & \end{pmatrix}$, $b = \begin{pmatrix} x^2 + y^2 \\ \vdots \end{pmatrix}$ and

$$t = \begin{pmatrix} r^2 - {x_0}^2 - {y_0}^2 \\ 2x_0 \\ 2y_0 \end{pmatrix}$$

A straight forward least squares approximation is then performed

$$t = (A^T A)^{-1} A^T b \tag{10}$$

where $t$ is used to solve for the estimated parameters $x_0, y_0$ and $r$.

A sliding window is used to select the points that are used for the parameter estimation. This approach was chosen to reduce computation and to ensure that up to date data is used.

## B. Control Strategies

There are three basic control strategies involved. Compliant motion (Section V-B.1), the lowest level of control, is used to correct the motion when errors occur. The predictive control (Section V-B.2) comes into play when there is a reliable estimate of the door parameters available. The last control strategy used is the configuration control (Section V-B.3), which drives the system towards a desired configuration, e.g. avoiding singularities.

### B.1 Compliant Motion

The wrist mounted force/torque sensor is used to perform relaxation of the end-effector. The manipulator is velocity controlled in Cartesian space which means that a simple relaxation of forces and torques can be written as

$$\dot{X}^{EE}_{compl} = K_{compl} F^{EE} \tag{11}$$

where $\dot{X}^{EE}$ is a 6 dimensional vector which describes the desired translational and angular velocities, $F^{EE}$ are forces and torques in 6 dimensions and $K_{compl}$ is a diagonal matrix which holds the corresponding gains.

It is assumed that in conjunction with the configuration control, the relaxation of the arm only introduces a small drift from the ideal configuration, thus singularities need not be considered here. The relaxation of the end-effector constitutes the lowest level of control in the system.

### B.2 Predictive Control

The predictive controller uses the model of the door with the currently estimated parameters to calculate a velocity vector in base coordinates that the platform should have in order to maintain end-effector motion along the arc.

This velocity vector is calculated by transforming the corresponding velocity at the end-effector down to the base coordinate system according to Equation 12, where $R^{EE}$ is the rotation of the end-effector relative to the base.

$$\dot{X}^{Base}_{pred} = \begin{bmatrix} R^{EE} & 0 \\ 0 & R^{EE} \end{bmatrix} \dot{X}^{EE}_{pred} \tag{12}$$

$\dot{X}^{EE}_{pred}$ is tangential to the arc that the door handle will follow, with a magnitude such that the door is opened with the desired speed (specified beforehand by the user).

### B.3 Configuration Control

The compliant control described above in Section V-B.1 introduces changes in the manipulator configuration. In order to keep the manipulator close to an ideal configuration, a mechanism is needed to simultaneously move the base and the end-effector such as to change the configuration of the arm while keeping the end-effector stationary relative to world coordinates.

The user has specified a desired configuration of the arm, expressed as a transform $T^{Base}_{desired}$. The current transform of the arm is written as $T^{Base}_{current}$. The objective now is to find a small transformation $\Delta T$ that can be applied to $T^{Base}_{current}$ as a way to move towards the desired configuration.

$$T^{Base}_{desired} = \Delta T \, T^{Base}_{current} \tag{13}$$

$\Delta T$ is converted to a six dimensional velocity vector $\dot{X}^{Base}_{error}$. A simple proportional control is applied:

$$\dot{X}^{Base}_{c} = K_{config} \dot{X}^{Base}_{error} \tag{14}$$

In Cartesian space the redundancy of the base and arm lies only in the x and y coordinates and in the rotation about the z-axis. Thus, we project the velocity onto these axes:

$$\dot{X}^{Base}_{config} = diag(1,1,0,0,0,1) \dot{X}^{Base}_{c} \tag{15}$$

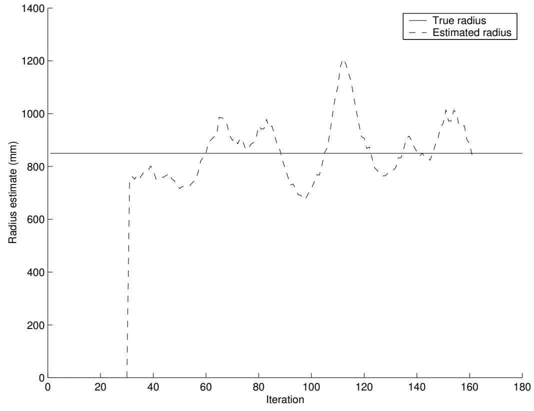



Fig. 11. Experimental results of radius estimation algorithm.

B.4 Control Law

The above strategies are integrated into the following control laws, for the arm:

$$\dot{X}^{Base}_{arm} = \begin{bmatrix} R^{EE} & 0 \\ 0 & R^{EE} \end{bmatrix} \dot{X}^{EE}_{compl} - \dot{X}^{Base}_{config} \quad (16)$$

and for the base:

$$\dot{X}^{Base}_{base} = \dot{X}^{Base}_{pred} + \dot{X}^{Base}_{config} \quad (17)$$

Note that the last terms of these two equations cancel, leading to configuration change but no end-effector motion.

### C. Experiments

The algorithm described above was implemented and tested using the parameters $K_{compl} = diag(0.0002, 0.0002, 0.0003, 0.00002, 0.00002.0.00002)\ ms^{-1}N^{-1}$ , $K_{config} = 0.005$ and with desired opening speed of 50mm/s. The performance of the opening algorithm was satisfactory and quite robust, opening the door approximately 90% of the time, neglecting failures in other parts of the integrated system mentioned in Section VI.

Figure 10 presents a comparison of the performance of the system with (a) all of the above strategies, (b) radius estimation only and (c) configuration management only. Note that the compliant motion was always used because compliance is required for real-world manipulation. First note that only case (a), with all components, succeeds in opening the door. Cases (b) and (c) both fail due to reaching singularities. In (b) the base moves too fast, following the predictions from the radius estimate and in (c) the robot has no radius estimate and assumes an infinite radius and so moves too far in the negative x direction.

Figure 11 shows that the on-line estimation of the door radius is fairly noisy. However, the estimate is perfectly adequate for opening the door. One reason for the noise is that we only use a window of the latest 30 points to estimate the radius. This corresponds to an arc of only 6°. We chose to use only 30 points so that the estimate would not be delayed too much.

## VI. Experiments

In Section IV-C we discussed the experiments from the computer vision point of view and in Section V-C we discussed the manipulation experiments and the results of the door parameter estimation. However, one of the objectives of this study has been to build an integrated system and so in this section we will discuss the experiments with the system a whole.

The procedures for door opening have been integrated with other components developed for the same robot testbed, such as a localisation system [21] and navigation and command system [6]. This means that the robot is able to perform navigation tasks and, in the event that a closed door is encountered, seamlessly start the door opening procedure described above.

However, the integrated system has not achieved the reliability that was desired. We have repeated the door opening procedure 24 times with the following results:

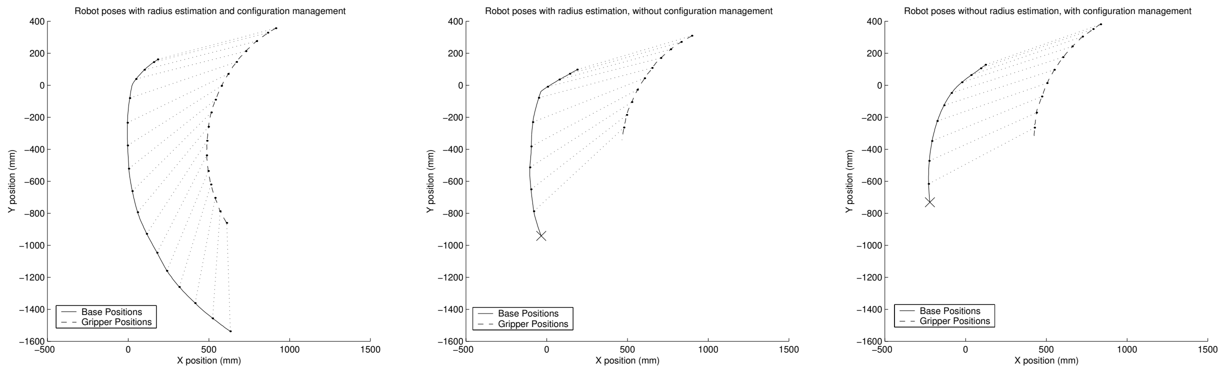
| | |
|---|---|
| Successfully opened door | 16 |
| Lost grip of door handle | 3 |
| Gripper failure/jam | 2 |
| Software failure | 1 |
| Laser sensor failure | 1 |
| Force sensor failure | 1 |

It appears that the cases where the robot lost it's grip on the door handle (through sliding off) are caused by errors in the visual servoing algorithm which can sometimes track vertical line structures on the door instead of the metal plate behind the handle. However, this problem is exacerbated by the fact that a parallel jaw gripper is not particularly suited for grasping a door handle and sliding motion requires only quite low force errors.

## VII. Discussion

We have presented the design and implementation of an integrated door opening system using a hybrid dynamic system model. Visual servoing is combined with compliant mobile manipulation to produce a system capable of performing a high-level, real-world task. The results presented here demonstrate that relatively simple algorithms from each field can be combined to produce quite effective high-level strategies. The results demonstrate the effectiveness of this approach and clearly indicate the value of higher level modeling. The system presented here is relatively straightforward in design but is very effective for solving a complex real-world task. It can never be said too often that the fusion of multiple sensors results in much more robust perception of the environment and the system implemented here is a clear example of this. In each case, the appropriate sensors were fused to obtain the required information.

The door opening system presented here is better than the prior work [4] (neglecting the undocumented ROMAN [3]) because it is a general solution which estimates the door position and radius on-line. In addition, the system presented here explicitly manages the redundant degrees of freedom of the mobile manipulator which [4] does not.




(a) (b) (c)

Fig. 10. Experimental results showing base and gripper motions using (a) both radius estimation and configuration management, (b) radius estimation only and (c) configuration management only. In all these cases the leftmost curve is the base position, the rightmost curve is the gripper position and the connecting lines indicate corresponding points on each curve. In (b) and (c) the cross marks failure due to singularity.

One obvious result of this study is that much more attention needs to be paid to the areas of systems integration and reliability in the field of robotics. It is obvious that systems integration takes a great deal of effort and future research towards reducing this effort is required. However, a less predictable result of this study is that there are a large number of low-level technical failures in the hardware and sensors that are available. This problem is multiplied when a large number of sensors are combined (a 1% failure rate in each component rapidly leads to an unusable system). Clearly, sensor fusion is a key issue and dependence on a single sensor (such as a force sensor) must be avoided. In this study, we have (through much programming effort) been largely able to overcome software issues but still the reliability of sensors and the gripper plagued us.

If robots are ever to be used in a home environment for real-world tasks, they will be amongst the most complex software and sensor-actuator systems in the world. If we look at other complex software projects (e.g. Windows 2000) or sensor-actuator systems (e.g. NASA's space program) we can see that a great deal of work is required in both of these areas.

## Acknowledgements

This work was sponsored in part by the Swedish Foundation for Strategic Research through its Centre for Autonomous Systems at KTH.

## References

[1] J. J. Craig, *Introduction to Robotics: Mechanics and control*, Addison-Wesley, 1989.

[2] O. Khatib, "Mobile manipulation: The robotic assistant," *Journal of Robotics and Autonomous Systems*, vol. 26, pp. 175–183, 1999.

[3] U. Hanebeck, C. Fischer, and G. Schmidt, "Roman: A mobile robotic assistant for indoor service applications," in *IROS-97*, Grenoble, June 1997, IEEE, vol. vol. 2, pp. 518–525, CS Press.

[4] K. Nagatani and S. Yuta, "Designing strategy and implementation of mobile manipulator control system for opening door," in *Proc. of International Conference on Robotics and Automation*, 1996, vol. 3, pp. 2828–2834.

[5] L. Petersson, M. Egerstedt, and H.I. Christensen, "A hybrid control architecture for mobile manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Kyongju, Korea, October 1999, IEEE, vol. 3, pp. 1285–1291.

[6] M. Andersson, A. Oreback, M. Lindstrm, and H. Christensen, "Isr: An intelligent service robot," in *Intelligent Sensor Based Robotics*, 1999, Lecture Notes in Artificial Intelligence, Heidelberg: Springer Verlag (To appear).

[7] R. Brockett, "Hybrid models for motion control systems," in *Essays on Control: Perspectives in the Theory and its Applications*. 1993, pp. 29–53, Birkhauser Boston.

[8] D. Kragić and H. I. Christensen, "A framework for visual servoing tasks," Acccepted for International Conference on Intelligent Autonomous Systems, 2000.

[9] L. Petersson, D. Austin, D. Kragić, and H. I. Christensen, "Towards an intelligent service robot system," Acccepted for International Conference on Intelligent Autonomous Systems, 2000.

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

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

[12] D. Terzopoulos and R. Szeliski, "Tracking with Kalman snakes," in *Active Vision*, A. Blake and A. Yullie, Eds., pp. 3–20. MIT Press, Cambridge, MA, 1992.

[13] C. Lee, "A comparison of two evidental reasoning schemas," *Artificial Intelligence*, vol. 35, no. 1, pp. 127–134, 1988.

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

[15] A.M. Nazif and M.D. Levine, "Low level image segmentation: an expert system," *IEEE Trans. PAMI*, vol. 6, no. 5, pp. 555–577, 1984.

[16] B. Parhami, "Voting algorithms," *IEEE Transactions on Reliability*, vol. 43, no. 3, pp. 617–629, 1994.

[17] P. Pirjanian, H. I. Christensen, and J.A. Fayman, "Application of voting to fusion of purposive modules: an experimental investigation," *Robotics and Autonomous Systems*, vol. 23, no. 4, pp. 253–266, 1998.

[18] D.M. Blough and G.F. Sullivan, "Voting using predispositions," *IEEE Transactions on reliability*, vol. 43, no. 4, pp. 604–616, 1994.

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

[20] J.A. Benediktsson, J.R. Sveinsson, and P.H. Swain, "Hybrid consensus theoretic classification," *IEEE Transactions on geoscience and remote sensing*, vol. 35, no. 4, pp. 833–842, 1997.

[21] Patric Jensfelt, David J. Austin, Olle Wijk, and Magnus Andersson, "Feature based condensation for mobile robot localization," in *IEEE Intl. Conf. on Robotics and Automation*, 2000.

**Lars Petersson** Lars Petersson received his M.Sc. degree in Engineering Physics from the Royal Institute of Technology in Stockholm, Sweden in 1996. From then on, he has been pursuing a Ph.D. in computer science at the Computational Vision and Active Perception Lab within the Centre for Autonomous Systems at the Royal Institute of Technology. His research interests are in the areas of mobile manipulation, sensor fusion and systems integration.

**David Austin** David Austin was born in Canberra, Australia, in 1973. He received his Ph.D. degree in Engineering from the Australian National University in 1999 (thesis titled Discrete Event Control of Uncertain Constrained Motion Systems). Since March 1999, he has been with the Centre for Autonomous Systems, Royal Institute of Technology, Stockholm, Sweden. His current interests are in the areas of hybrid dynamic systems, manipulation and mobile robot map building.

**Danica Kragić** Danica Kragić received the M.Sc. degree in Mechanical Engineering from Technical University of Rijeka, Croatia in 1995. She is currently pursuing the Ph.D. in computer science at the Computer Vision and Active Perception Lab and Centre for Autonomous Systems at the Royal Institute of Technology, Sweden. Her research interests include real–time vision, visual servoing and data fusion.