ICT Call 7
ROBOHOW.COG
FP7-ICT-288533

Deliverable D4.1:

Virtual visual servoing for tool calibration

April 15, 2014

Project acronym:	ROBOHOW.COG
Project full title:	Web-enabled and Experience-based Cognitive Robots that Learn Complex Everyday Manipulation Tasks
Work Package:	WP 4
Document number:	D4.1
Document title:	Virtual visual servoing for tool calibration
Version:	1.0
Delivery date:	January 31st, 2013
Nature:	Report
Dissemination level:	Public
Authors:	Christian Smith (KTH) Yiannis Karayiannidis (KTH) Danica Kragic (KTH)

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n^{o}288533 ROBOHOW.COG.

Contents

Summary

The contribution in this deliverable is on perception of manipulated objects, as detailed in Task 4.3. When an object is manipulated by a robot, its pose in the robot hand may not be easily estimated due to the occlusion or abrupt changes in the pose. Estimating the exact pose of the object is necessary for successfully performing subsequent tasks in which the objects is used as a tool.
We realize perceptual capabilities that will calibrate the tools with respect to the hand after they have been grasped properly such that the robot can compute the accurate pose of the tool actuator based on the joint angles of the robot arm and hand. To this end, algorithms for tracking of known objects that undergo motion changes during grasping have been developed.

Chapter 1
Published Results

The results for this deliverable have been accepted for publication in peer-reviewed venues. This section contains a short description of the contributions, and references to the published reports, which are appended to this document.

1.1 Virtual Visual Servoing

We study visual servoing in a framework of detection and grasping of unknown objects. Classically, visual servoing has been used for applications where the object to be servoed on is known to the robot prior to the task execution. In addition, most of the methods concentrate on aligning the robot hand with the object without grasping it. In our work, visual servoing techniques are used as building blocks in a system capable of detecting and grasping unknown objects in natural scenes. We show how different visual servoing techniques facilitate a complete grasping cycle. This work was initiated in FP7 project GRASP (IST-FP7-IP-215821), and has been continued in RoboHow. The results are published in [1].

1.2 Kinematic Estimation

We address the problem of robot interaction with mechanisms with joints that afford a constrained motion in one degree of freedom, such as doors, drawers and cupboards. We propose a methodology for the simultaneous operation of the mechanism and estimation of constraints imposed by the joint, that requires no prior knowledge of the objects' kinematics, including the type of joint - either prismatic or revolute. The method consists of a velocity controller which relies on force/torque measurements and estimation of the motion direction, rotational axis and the distance from the center of rotation in the case of a hinged door. The results are published in [2, 3, 4, 5].

Bibliography

[1] X. Gratal, J. Romero, J. Bohg, and D. Kragic, "Visual servoing on unknown objects," *Mechatronics*, vol. 22, no. 4, pp. 423–435, 2012.

[2] Y. Karayiannidis, C. Smith, P. Ögren, and D. Kragic, "Adaptive force/velocity control for opening unknown doors," in *International IFAC Symposium on Robotic Control*, 2012.

[3] Y. Karayiannidis, C. Smith, F. Vina, P. Ögren, and D. Kragic, ""open sesame!" - adaptive force/velocity control for opening unknown doors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 4040–4047.

[4] ——, "Model-free robot manipulation of doors and drawers by means of fixed-grasps," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[5] ——, "Design of force-driven online motion plans for door opening under uncertainties," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on Real-time Motion Planning: Online, Reactive, and in Real-time*, 2012.

Mechatronics 22 (2012) 423–435

Contents lists available at SciVerse ScienceDirect

Mechatronics

journal homepage: www.elsevier.com/locate/mechatronics

Visual servoing on unknown objects

Xavi Gratal *, Javier Romero, Jeannette Bohg, Danica Kragic

Computer Vision and Active Perception Lab, Centre for Autonomous Systems, School of Computer Science and Communication, Royal Institute of Technology, 10044 Stockholm, Sweden

ARTICLE INFO

Article history:
Available online 10 November 2011

Keywords:
Visual servoing
Object grasping
Calibration
Active vision

ABSTRACT

We study visual servoing in a framework of detection and grasping of unknown objects. Classically, visual servoing has been used for applications where the object to be servoed on is known to the robot prior to the task execution. In addition, most of the methods concentrate on aligning the robot hand with the object without grasping it. In our work, visual servoing techniques are used as building blocks in a system capable of detecting and grasping unknown objects in natural scenes. We show how different visual servoing techniques facilitate a complete grasping cycle.

1. Introduction

Object grasping and manipulation stands as an open problem in the field of robotics. Many approaches assume that the object to be manipulated is known beforehand [1–4]. If the object bears some resemblance to a known object, experience can be used for grasp synthesis [5–8]. An unknown object needs to be analyzed in terms of its 3D structure and other physical properties from which a suitable grasp can be inferred [9–12].

Realistic applications require going beyond open-loop execution of these grasps and the ability to deal with different type of errors occurring in an integrated robotic system. One kind of errors are systematic and repeatable, introduced mainly by inaccurate kinematic models. These can be minimized offline through precise calibration. The second kind are random errors introduced by a limited repeatability of the motors or sensor noise. These have to be compensated for through online mechanisms.

In this paper, we show how *Visual Servoing* (VS) can be used at different stages in a grasping pipeline to correct errors both offline and online. One of the contributions is the application of VS for automatic offline calibration of the hardware. During online execution, we follow the classical approach of applying VS for aligning the robot hand with the object prior to grasping it [13]. This requires tracking of the manipulator pose relative to the camera. Instead of the common approach of putting markers on the robot hand, we use a model based tracking system. This is achieved through *Virtual Visual Servoing* (VVS) in which the systematic and random errors are compensated for. An additional contribution is the generalisation of VVS to CAD models instead of using object models which are tailor-made for the application.

The remainder of this paper is organised as follows. Section 2 formalises the systematic and random errors inherent to the different parts of the robotic system. In Section 3, related work in the area of offline calibration as well as closed-loop control is presented. The approach proposed in this paper is described in detail in Section 4. Its performance is analysed quantitatively on synthetic data and qualitatively on our robotic platform. The results of these experiments are presented in Section 5.

2. Problem formulation

Object grasping in real world scenarios requires a set of steps to be performed prior to the actual manipulation of an object. A general outline of our grasping pipeline is provided in Fig. 2. The hardware components of the system as shown in Fig. 1 are (i) the Armar III robotic head [14] equipped with two stereo camera pairs (wide-angle for peripheral vision and narrow-angle for foveal vision), (ii) the 6 DoF Kuka arm KR5 sixx R850 [15] and (iii) the Schunk dexterous hand 2.0 (SDH) [16].

2.1. Grasping pipeline

The main pre-requisite for a robot to perform a pick-and-place task is to have an understanding of the 3D environment it is acting in. In our pipeline, we perform scene construction by using the active head for visual exploration and stereo reconstruction as described in detail in our previous work [17]. The resulting point cloud is segmented into object hypotheses and background.

The scene model then consists of the arm and the active head positioned relative to each other based on the offline calibration as described in Section 4.4. Furthermore, a table plane is detected and the online detected object hypotheses are placed on it.

* Corresponding author.
E-mail addresses: javiergm@nada.kth.se (X. Gratal), jrgn@nada.kth.se (J. Romero), bohg@nada.kth.se (J. Bohg), danik@nada.kth.se (D. Kragic).

0957-4158/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mechatronics.2011.09.009

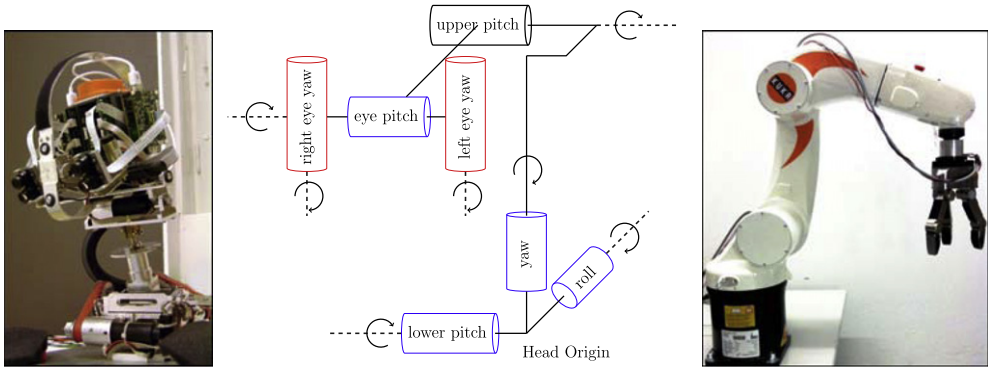


Fig. 1. Hardware components in the grasping pipeline. (a) Armar III active head with 7 DoF. (b) The kinematic chain of the active head. The upper pitch is kept static. Right and left eye yaw are actuated during fixation and thereby change the vergence angle and the epipolar geometry. All the other joints are used for gaze shifts. (c) Kuka arm with 6 DoF and Schunk dexterous hand 2.0 (SDH) with 7 DoF.

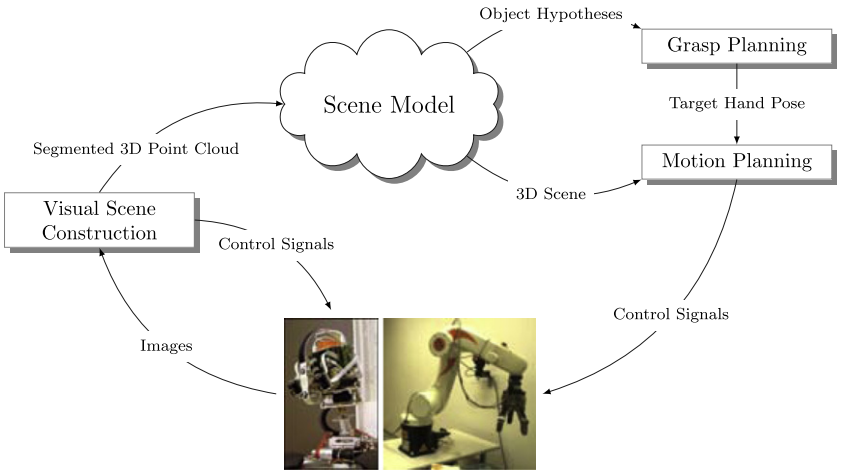


Fig. 2. Our open-loop grasping pipeline.

Grasp inference is then performed on each hypothesis. For some given grasp candidates, a collision-free arm trajectory is planned in the scene model and applied to the object hypothesis with the real arm.

2.2. Error formalisation

The aforementioned grasping pipeline relies on the assumption that all the parameters of the system are perfectly known. This includes e.g. internal and external camera parameters as well as the pose of the head, arm and hand in a globally consistent coordinate frame.

In reality, however, two different types of errors are inherent to the system. One contains the systematic errors that are repeatable and arise from inaccurate calibration or inaccurate kinematic models. The other group comprises random errors originating from noise in the motor encoders or in the camera. These errors propagate and deteriorate the relative alignment between hand and object. The most reliable component of the system is the Kuka arm that has a repeatability of less than 0.03 mm [15]. In the following, we will analyse the different error sources in more detail.

2.2.1. Stereo calibration error

Given a set of 3D points $Q^W = [x^W y^W z^W]^T$ in the world reference frame W and their corresponding pixel coordinates $P = [u\, v 1]^T$ in the image, we can determine the internal parameters $\mathbf{C}$ and external parameters $[\mathbf{R}_W^C|\mathbf{t}_W^C]$ for all cameras C in the vision system. This is done through standard methods by exploiting the following relationship between Q^W and P:

$$wP = \begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \mathbf{C}P^C \quad \text{with } P^C = \begin{bmatrix} z^C \\ y^C \\ z^C \end{bmatrix} = \mathbf{R}_W^C \left[P^W - \mathbf{t}_W^C\right] \tag{1}$$

Once we have these parameters for the left camera L and right camera R, the epipolar geometry as defined by the essential matrix $\mathbf{E} = \mathbf{R}_L^R [\mathbf{t}_L^R]_\times$ can be determined. Here $\mathbf{t}_L^R$ defines the baseline and $\mathbf{R}_L^R$ the rotation between the left and right camera system.

Our calibration method, which uses the arm as world reference frame, will be described in Section 4.4. The average reprojection error is 0.1 pixels. Since peripheral and foveal cameras are calibrated simultaneously, we also get the transformation between them.

Additionally to the systematic error in the camera parameters, other effects such as camera noise and specularities lead to random errors in the stereo matching.

2.2.2. Positioning error of the active head

We are using the robot head to actively explore the environment through gaze shifts and fixation on objects. This involves dynamically changing the epipolar geometry between the left and right camera system. Also the camera position relative to the head origin is changed. The internal parameters remain static.

The kinematic chain of the active head is shown in Fig. 1b. Only the last two joints, the left and right eye yaw, are used for fixation

and thereby affect the stereo calibration. The remaining joints are actuated for performing gaze shifts.

In order to accurately detect objects with respect to the camera, the relation between the two camera systems as well as between the cameras and the head origin needs to be determined after each movement. Ideally, these transformations should be obtainable just from the known kinematic chain of the robot and the readings from the encoders. In reality, these readings are erroneous due to noise and inaccuracies in the kinematic model.

Inaccuracies arise in the manufacturing process influencing the true center and axis of joint rotations and in the discrepancy between motor encoder readings and actual angular joint movement. This is illustrated in Fig. 3 showing the translational component $\delta = [\delta_x \delta_y \delta_z]^T$ and rotational component $\epsilon = [\epsilon_x \epsilon_y \epsilon_z]^T$ of the error between the ideal and real joint positions. Error matrices can be defined for every joint modelling inaccurate positioning and motion. Let us define J_{n-1} and J_n as two subsequent joints. According to the Denavit–Hartenberg convention, the ideal transformation $\mathbf{T}^n_{n-1}$ between these joints is defined as

$$\mathbf{T}^n_{n-1} = {}_z\mathbf{T}^n_{n-1}(d,\phi)\,{}_x\mathbf{T}^n_{n-1}(a,\alpha) \tag{2}$$

where ${}_z\mathbf{T}^n_{n-1}(d,\phi)$ describes the translation d and rotation ϕ with respect to the z-axis of J_{n-1}. ${}_x\mathbf{T}^n_{n-1}(a,\alpha)$ describes the translation a and rotation α with respect to the x-axis of J_n. While d, a and α are defined by the kinematic model of the head, ϕ is varying with the motion of the joints and can be read from the motor encoders.

The true transformation ${}_e\mathbf{T}^n_{n-1}$ will however look different. Defining the position error $\mathbf{T}_{pe}$ and motion error $\mathbf{T}_{me}$ as the transformation matrices associated to the translation error δ and rotation error $\boldsymbol{\epsilon}$ for the joint yields

$${}_e\mathbf{T}^n_{n-1} = {}_z\mathbf{T}^n_{n-1}(d,\phi)\mathbf{T}_{me}\;{}_x\mathbf{T}^n_{n-1}(a,\alpha)\mathbf{T}_{pe}. \tag{3}$$

These errors propagate through the kinematic chain and mainly affect the x and y position of points relative to the world coordinate system.

Regarding random error, the last five joints in the kinematic chain achieve a repeatability in the range of ±0.025° [14]. The neck pitch and neck roll joints in Fig. 1b achieve a repeatability in the range of ±0.13 and ±0.075 ° respectively.

2.2.3. Positioning error of the cameras with respect to the arm

As will be described in Section 4.4, we are using the arm to calibrate the stereo system. Therefore, assuming that the transformation from the head origin to the cameras is given, the error in the transformation between the arm and cameras is equivalent to the error of the stereo calibration.

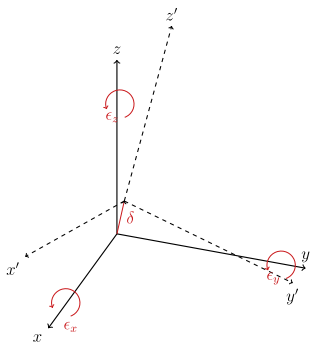


Fig. 3. Six parametric errors in a rotary joint around the z-axis. $\boldsymbol{\delta} = [\delta_x \delta_y \delta_z]^T$ is the translational component and $\boldsymbol{\epsilon} = [\epsilon_x \epsilon_y \epsilon_z]^T$ is the rotational component of the error.

3. Related work and contributions

3.1. Closed-loop control in robotic grasping

In the previous section, we summarised the errors in a grasping system that lead to an erroneous alignment of the end effector with an object. A system that executes a grasp in closed loop without any sensory feedback is likely to fail.

In [18,19] this problem is tackled by introducing haptic and force feedback into the system. Control laws are defined that adapt the pose of the end effector based on the readings from a force–torque sensor and contact location on the haptic sensors. A disadvantage of this approach is that the previously detected object pose might change during the alignment process.

Other grasping systems make use of visual feedback to correct the wrong alignment before contact between the manipulator and the object is established. Examples are proposed by Huebner et al. [2] and Ude et al. [20], who are using a similar robotic platform to ours including an active head. In [2], the Armar III humanoid robot is enabled to grasp and manipulate known objects in a kitchen environment. Similar to our system [17], a number of perceptual modules are at play to fulfill this task. Attention is used for scene search. Objects are recognized and their pose estimated with the approach originally proposed in [4]. Once the object identity is known, a suitable grasp configuration can be selected from an offline constructed database. Visual servoing based on a spherical marker attached to the wrist is applied to bring the robotic hand to the desired grasp position [21]. Different from our approach, absolute 3D data is estimated by fixing the 3 DoF for the eyes to a position for which a stereo calibration exists. The remaining degrees of freedom controlling the neck of the head are used to keep the target and current hand position in view. In our approach, we reconstruct the 3D scene by keeping the eyes of the robot in constant fixation on the current object of interest. This ensures that the left and right visual field overlap as much as possible, thereby maximizing e.g. the amount of 3D data that can be reconstructed. However, the calibration process becomes much more complex.

In the work by Ude et al. [20], fixation plays an integral part of the vision system. Their goal is however somewhat different from ours. Given that an object has already been placed in the hand of the robot, it moves it in front of its eyes through closed loop vision based control. By doing this, it gathers several views from the currently unknown object for extracting a view-based representation that is suitable for recognizing it later on. Different to our work, no absolute 3D information is extracted for the purpose of object representation. Furthermore, the problem of aligning the robotic hand with the object is circumvented.

3.2. Calibration methods

In [22], the authors presented a method for calibrating the active stereo head. The correct depth estimation of the system was demonstrated by letting it grasp an object held in front of its eyes. No dense stereo reconstruction has been shown in this work.

Similarly, in [23] a procedure for calibrating the Armar III robotic head was presented. Our calibration procedure is similar to the one described in those papers, with a few differences. We extend it to the calibration of all joints, thus obtaining the whole kinematic chain. Also, the basic calibration method is modified to use an active pattern instead of a fixed checkerboard, which has some advantages that we outline in Section 4.4.

In [24], a robot moves a checkerboard pattern in front of its cameras to several different poses. Similar to our approach, it can use its arms as kinematic chain sensors and automate the otherwise tedious calibration process. However, the poses of the robot

are hand crafted such that the checkerboard is guaranteed to be visible to the sensor that is to be calibrated. In our approach, we use visual servoing to automatically generate a calibration pattern that is uniformely distributed in camera space.

3.3. Visual and virtual visual servoing

For the accurate control of the robotic manipulator using visual servoing, it is necessary to know its position and orientation (*pose*) with respect to the camera. In the systems mentioned in Section 3.1, the end effectors of the robots are tracked based on fiducial markers like LEDs, colored spheres or Augmented Reality tags. A disadvantage of this marker-based approach is that the mobility of the robot arm is constrained to keep the marker always in view. Furthermore, the position of the marker with respect to the end effector has to be known exactly. For these reasons, we propose to track the whole manipulator instead of only a marker. Thereby, we alleviate the problem of constrained arm movement. Additionally, collisions with the object or other obstacles can be avoided in a more precise way. In this paper we track the pose of the robotic arm and hand assuming their kinematic chain to be perfectly calibrated, although the system can be adapted to estimate deviations in the kinematic chain.

Tracking objects of complex geometry is not a new problem and approaches can be divided into two groups: appearance-based and model-based methods. The first approach is based on comparing camera images with a huge database of stored images with annotated poses [25]. The second approach relies on the use of a geometrical model (3D CAD model) of the object and perform tracking based on optical flow [26]. There have also been examples that integrate both of these approaches [27].

Apart from the tracking itself, an important problem is the initialization of the tracking process. This can be done by first localizing the object in the image followed by a global pose estimation step. In our previous work, we have also demonstrated how the initialization can be done for objects in a generic way [28]. In the case of a manipulator, a rough estimate of its pose in the camera frame can be obtained from the kinematics of the arm and the hand-eye calibration. In [29], it has been shown that the error between this first estimate and the real pose of the manipulator can be corrected through virtual visual servoing, using a simple wireframe model of the object. In our work, a synthetic image of the robot arm is rendered based on a complete 3D CAD model and its initial pose estimate is compared and aligned with the real image.

In our recent work [30], we demonstrate how pose estimation can be performed for a complex articulated object such as a human hand. The major contribution of that work is the use of a discriminative machine learning approach for obtaining real-time tracking of an object with 27 degrees of freedom. The problem considered in this paper has a lower dimensionality and a more accurate guess of the initial pose. This makes it possible to adopt a real-time generative approach that renders the last pose of the object in each frame and estimates the new pose through a process of error minimization.

4. The proposed system

For overcoming the problem of lacking a globally consistent coordinate frame for objects, manipulator and cameras, we introduce visual servoing into the grasping pipeline. This allows us to control the manipulator in closed loop using visual feedback to correct any misalignment with the object online. We then use the camera coordinate frame as the global reference system in which the manipulated object and robot are also defined.

For accurately tracking the pose of the manipulator, we use virtual visual servoing. The initialisation of this method is based on the known joint values of hand and arm and the hand-eye calibration, which is obtained offline.

The adapted grasping pipeline is summarised in Fig. 4. What follows is a more detailed description of all the components of this pipeline.

4.1. Vision system for constructing the scene model

As described earlier, the scene model in which grasping is performed consists of a robot arm, hand and head as well as a table plane on which object hypotheses are placed. The emergence of these hypotheses is triggered by the visual exploration of the scene with the robotic head. In the following, we will give a brief summary of this exploration process. More details can be found in our previous work [3,32,33,17].

The active robot head has two stereo camera pairs. The wide-field cameras of which an example can be seen in Fig. 5a are used for scene search. This is done by computing a saliency map on them and assuming that maxima in this map are initial object hypotheses (Fig. 5b). More details on the saliency map which is based on the Itti & Koch model [34] can be found in [3]. A gaze shift is performed to a maximum such that the stereo camera with the narrow-angle lenses center on the potential object. An example of an object fixated in the foveal view is shown in Fig. 5c. In the following we will label the camera pose when fixated on the current object of interest as C_0. Once the system is in fixation, a disparity map is calculated and segmentation performed (see Fig. 5e and d). For each object, we then obtain a 3D point cloud (an example is shown in Fig. 5f).

4.2. Grasp and motion planning

Once a scene model has been obtained by the vision system, a suitable grasp can be selected for each object hypothesis depending on the available knowledge about the object. In our previous work, we presented grasp planners on known, familiar or unknown objects [2,5,31,17]. In [2,31] resulting grasp hypotheses are tested in simulation on force closure prior to execution. The set of stable grasps is then returned to plan corresponding collision-free arm trajectories.

The focus of this paper is the application of visual servoing and virtual visual servoing in the grasping pipeline. We have therefore selected a simple top-grasp selection mechanism that has been proven to be very effective for pick-and-place tasks in table-top scenarios [17]. For this, we calculate the eigenvectors and centre of mass of the projection of the point cloud on the table plane. An example of such a projection can be seen in Fig. 5f (left). The corresponding grasp is a top grasp approaching the centre of mass of this projection. The wrist orientation of the hand is determined such that the vector between fingers and thumb is aligned with the minor eigenvector. A grasping point $G^{C_0} = [zx\ zy\ z]$ in camera space C_0 is then formed with the x and y coordinates of the center of the segmentation mask as obtained during fixating on the object. The depth of this point, i.e., the z coordinate is computed from the vergence angle as read from the motor encoders. The goal is then to align the tool center point of the end effector with this grasping point. Using more sophisticated grasp planners that can deal with more complex scenarios is regarded as future work.

4.3. Object localisation in different viewing frames

The robotic head moves during the grasping process for focusing on different parts of the environment (e.g. different objects, the robotic hand and arm). In each of these views the object to

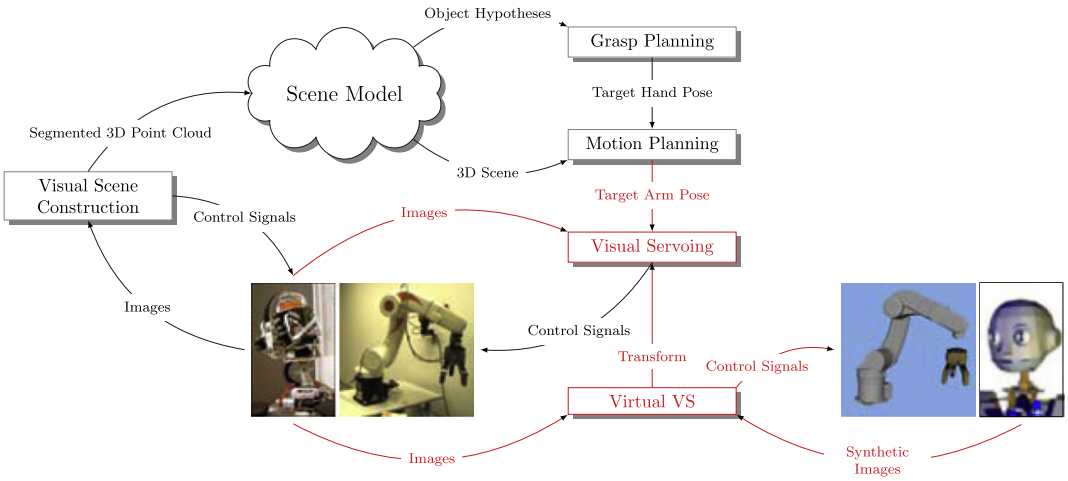


Fig. 4. A grasping pipeline with closed-loop control of the manipulator through visual servoing that is initialised through virtual visual servoing. Armar III head model adapted from [31].

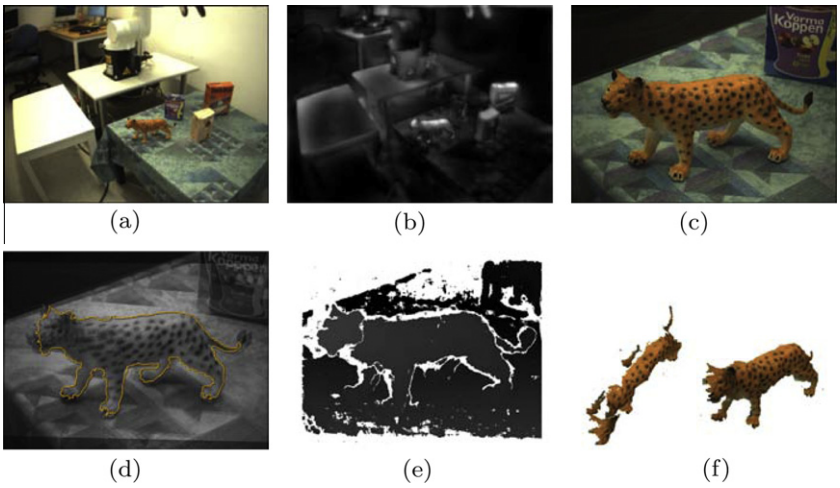


Fig. 5. Example output for exploration process. (a) Left peripheral camera. (b) Saliency map of peripheral image. (c) Left foveal camera. (d) Segmentation on overlayed fixated rectified left and right images. (e) Disparity map. (f) Point cloud of tiger.

be grasped should remain localized relative to the current camera frame C_k to accurately align the end effector with it. The new position of the object can be estimated given the new pose of the head, which is determined based on the motor reading and the forward kinematics of the head as depicted in Fig. 1b. However, we cannot completely rely on this estimate due to inaccuracies in the kinematic model and motor encoders. For this reason we perform a local refinement of the object position in the new view of the scene based on template matching.

A template is generated for each object when the head is fixated on it. Based on the segmentation mask in the foveal view as shown in Fig. 5d and the calibration between the peripheral and foveal cameras, we generate a tight bounding box around the object in the peripheral view. Each time the head moves, the new position of this template can be estimated based on the old and new pose of the head. We create a window of possible positions of the object by growing this initial estimate by a fixed amount of pixels. We perform a sliding window comparison between all possible locations of the object template within this window and the object template. The location which returns the lowest mean square error is the one suggested as x and y coordinates of the object position in the new view.

4.4. Offline calibration process

In Section 2.2, we analysed the errors that got introduced by the different parts of the system. The group of systematic errors can be minimized by offline calibration. In the following, we will introduce our method for stereo and hand-eye calibration as well as the calibration of the kinematic chain of the robotic head.

4.4.1. Stereo calibration

One of the most commonly used methods for finding the transformation between two camera coordinate systems is the use of a checkerboard which is observed by two cameras (or the same camera before and after moving) [35]. The checkerboard defines its own coordinate system in which the corners of the squares are the set of 3D points Q^A in the arm coordinate system A as in Section 2.2.1. The detection of these corners in the left and right images gives us the corresponding pixel coordinates P from which we can solve for the internal and external camera parameters. Given these, we can obtain the transformation between the left and right camera coordinate frames.

We used a modified version of this method. Instead of a checkerboard pattern, we used a small LED rigidly attached to the end

effector of the robotic arm, which we can detect in the image with subpixel precision. Because of the accuracy and repeatability of the KUKA arm (<0.3 mm), we can move the LED to a number of places for which we know the exact position in arm space, which allows us to obtain the transformation between arm and camera space.

This method has several advantages over the use of a traditional checkerboard pattern:

- Instead of an arbitrary checkerboard coordinate system as intermediate coordinate system, we can use the arm coordinate system. In this way we are obtaining the hand-eye calibration for free at the same time that we are performing the stereo calibration.
- In the checkerboard calibration method, the checkerboard ought to be fully visible in the two calibrating cameras for every calibration pose. This may be difficult when the two camera poses are not similar or their field of view is small. With our approach, it is not necessary to use exactly the same end effector positions for calibrating the two cameras since all points are defined in the static arm coordinate system that is always valid independently of camera pose.
- For these same reasons, we found empirically that this approach makes it possible to choose a pattern that offers a better calibration performance. For example, by using a set of calibration points that is uniformly distributed in camera space (as opposed to world space, which is the case for checkerboard patterns), it is possible to obtain a better characterization of the lenses distortion parameters.

The main building block for this system is a visual servoing loop that allows us to bring the LED to several predefined positions in camera space. In this loop, we want to control the position of the LED (3 DOF) using only its projection in the image (2 DOF). Therefore, we introduce an additional constraint by limiting the movement of the LED to a predefined plane in arm space.

We define S_0^A as a point on this plane and S^A as its normal vector. Additionally, we need a rough estimate of the arm to camera coordinate transformation $\boldsymbol{A}_A^C$, and of the camera matrix $\boldsymbol{C}$. They are defined as follows:

$$\boldsymbol{A}_A^C = \left[\boldsymbol{R}_A^C | \boldsymbol{t}_A^C\right], \quad \boldsymbol{C} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

We can then find the plane in camera space:

$$S_0^C = \boldsymbol{R}_A^C S_0^A + \boldsymbol{t}_A^C \quad S^C = \boldsymbol{R}_A^C S^A \tag{5}$$

The process of moving the LED to a position in the image is then as follows: we define a target point in the image $P_t = (u_t, v_t)$ (specified in pixels) where we want to bring the LED. For each visual servoing iteration we detect the current position $P_c = (u_c, u_c)$ (again in pixels) of the LED in the image. We can then use the camera matrix to convert these points to homogeneous camera coordinates:

$$P_t^C = \boldsymbol{C}^{-1}[x_t \; y_t \; 1]^T = \left[\frac{1}{f}x_t \; \frac{1}{f}y_t \; 1\right]^T \tag{6}$$

$$P_c^C = \boldsymbol{C}^{-1}[x_c \; y_c \; 1]^T = \left[\frac{1}{f}x_c \; \frac{1}{f}y_c \; 1\right]^T \tag{7}$$

Then, we can project these homogeneous points into the plane defined by S_0^C and S^C. A point Q is in that plane if $Q \cdot S^C = S_0^C \cdot S^C$. Therefore, the projection of the homogeneous points P_t^C and P_c^C into the plane can be found as

$$Q_t^C = \frac{S_0^C \cdot S^C}{P_t^C \cdot S^C} P_t^C \qquad Q_c^C = \frac{S_0^C \cdot S^C}{P_c^C \cdot S^C} P_c^C \tag{8}$$

From these, we can obtain the vector

$$d^C = Q_t^C - Q_c^C \tag{9}$$

which is the displacement from the current to the target LED positions in camera space, and then

$$d^A = \boldsymbol{R}_A^{C^{-1}} d^C \tag{10}$$

which is the correspondent displacement in arm space.

We can easily see that the displacement obtained in arm space is within the given plane since

$$d^C \cdot S^C = \left(Q_t^C - Q_c^C\right) \cdot S^C = Q_t^C \cdot S^C - Q_c^C \cdot S^C = 0 \tag{11}$$

$$d^A \cdot S^A = d^{A^T} S^A = \left(\mathbf{R}_A^{C^{-1}} d^C\right)^T \left(\mathbf{R}_A^{C^{-1}} S^C\right) = d^{C^T} \mathbf{R}_A^C \mathbf{R}_A^{C^{-1}} s^C = d^C \cdot S^C = 0 \tag{12}$$

We can then use this displacement to obtain a simple proportional control law

$$\boldsymbol{u} = k d^A \tag{13}$$

where $\boldsymbol{u}$ is the velocity of the end effector and k is a constant gain factor.

This control is then used to update the position in every iteration until convergence, i.e. the arm has reached the desired position with an error below a certain threshold.

Algorithm 1. Pseudo code for defining the calibration pattern

Data: Number of points n on each depth plane, number of depth planes m
Result: Set of Correspondences $\left[Q_i^A, P_i\right]$ with $(0 < i \leqslant m \cdot n)$
begin
// `Initialising the principal axis of the camera in arm space`
// `VisualServoing(P,S) is a function that brings the LED to`
// `the point P in image space within the plane S in arm space`
$\mathbf{S}_0$ = Some plane at a distance d_0 from the camera
C_0^A = `VisualServoing` $(0,0)$, $\mathbf{S}_0$
$\mathbf{S}_1$ = Some plane at a distance d_1 from the camera
C_1^A = `VisualServoing` $(0,0)$, $\mathbf{S}_1$
$d = (C_1^A - C_0^A)/(m-1)$
$i = 0$
foreach $l \in [0 \ldots m-1]$ **do**
  $\mathbf{S}$ = plane with normal d which contains $C_0^A + ld$
  foreach P_k *with* $(0 < k \leqslant n)$ **do**
    Q_i^A = `VisualServoing` (x_k, x_y),$\mathbf{S}$
    $i{+}{+}$
  end
end
end

Once we have the system which allows us to bring the LED to a certain position in the image we can start generating our calibration pattern. First, we bring the LED to the center of the image in two parallel planes, which are located at different distances from the camera, and record their positions C_0^A and C_1^A in arm space. From this, we can obtain the vector $S^A = C_0^A - C_1^A$ which is parallel

to the principal axis of the camera. Any plane perpendicular to this vector will be parallel to the image plane. We can then bring the LED to some fixed positions along these planes, thus generating points which are uniformly distributed both in the image and in depth (by using planes which are separated by a constant distance). The generated pattern looks like the one shown in Fig. 6a. Algorithm 1 shows the method in more detail. In our system, we used 6×6 points rectangular patterns in 6 different depths, for a total of 216 calibration points. With this, we achieved an average reprojection error of 0.1 pixels.

4.4.2. Head-eye calibration

For a static camera setup, the calibration process would be completed here. However, our vision system can move to fixate on the objects we manipulate. Due to inaccuracies in the kinematic model of the head, we cannot obtain the exact transformation between the camera coordinate system before and after moving a certain joint from the motor encoders.

In Section 2.2.2, we described the transformation error that arises from the misalignment of the real and ideal coordinate frame of a joint. In our method, we are minimizing the effect of this error by finding the true position and orientation of the real coordinate frame as follows:

1. Choose two different positions of the joint, that are far enough apart to be significant, but with an overlapping viewing area that is still reachable for the robotic arm.
2. For each of these two positions, perform the static calibration process as described above, so that we obtain the transformation between the arm coordinate system and each of the camera coordinate systems.
3. Find the transformation between the camera coordinate systems in the two previously chosen joint configurations. This transformation is the result of rotating the joint around some roughly known axis (it is not precisely known because of mechanical inaccuracies), with a roughly known angle from the motor encoders. From the computed transformation, we can then more exactly determine this axis, center and angle of rotation.

This is illustrated in Fig. 6b.

In our case, the results showed that while the magnitude of the angles of rotation differed significantly from what could be obtained from the kinematic chain, the orientation and position of these axes were quite precise in the specifications. To avoid overly complicating the model, we decided to correct only the actual angles α in Eq. (2), except for the vergence joint. For this joint, the orientation was also corrected, since here small errors have a large impact in the accuracy of depth estimation.

4.5. Virtual visual servoing

As mentioned in Sections 1 and 3.3, our system uses virtual visual servoing to refine the position of the arm and hand in camera space provided by the calibration system. In Fig. 8, the difference between the estimated arm pose and the corrected one is visualized. In this section we will formalize the problem and explain how we solved it.

The pose of an object is denoted by $\mathbf{M}(\mathbf{R},\mathbf{t})$ where $\mathbf{t} \in \mathcal{R}(3), \mathbf{R} \in \mathbf{SO}(3)$. The set of all poses that the robot's end-effector can attain is denoted with $\mathcal{T}_G \subseteq \mathbf{SE}(3) = \mathcal{R}(3) \times \mathbf{SO}(3)$.

Pose estimation considers the computation of a rotation matrix (orientation) and a translation vector of the object (position), $\mathbf{M}(\mathbf{R},\mathbf{t})$:

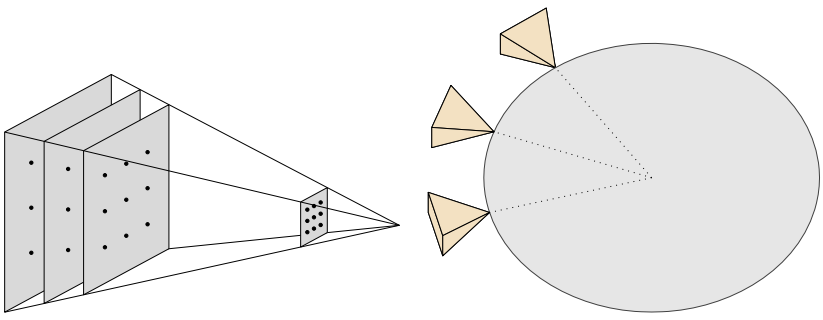
$$\mathbf{M} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_X \\ r_{21} & r_{22} & r_{23} & T_Y \\ r_{31} & r_{32} & r_{33} & T_Z \end{bmatrix} \tag{14}$$

The equations used to describe the projection of a three-dimensional model point $\mathbf{Q}$ into homogeneous coordinates of the image point $[xy]^T$ are:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \mathbf{R}[\mathbf{Q} - \mathbf{t}] \quad \text{with } [wx, wy, w] = \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{15}$$

where $\mathbf{P}$ represents the internal camera parameters matrix including focal length and aspect ratio of the pixels, w is the scaling factor, $\mathbf{R}$ and $\mathbf{t}$ represent the rotation matrix and translation vector, respectively.

Our approach to pose estimation and tracking is based on virtual visual servoing where a rendered model of the robot parts is aligned with the current image of their real counterparts. The outline of the system is presented in Fig. 7. In order to achieve the alignment, we can either control the position of the part to bring it to the desired pose or move the *virtual* camera so that the *virtual* image corresponds to the current camera image, denoted as *real* camera image in the rest of the paper. Using the latter approach has the problem that the local effect of a small change in orientation of the camera is very similar to a large change in its position, which leads to convergence problems. Therefore, in this paper, we adopt the first approach where we render synthetic images by incrementally changing the pose of the tracked part. In the first iteration, the position is given based on the forward kinematics. Then, we extract visual features from the rendered image, and compare them with the features extracted from the current camera image. The details about the features that are extracted are given in Section 4.5.2.



(a) Movement pattern of the end effector for offline calibration.

(b) Three viewing directions of a camera rotated around one axis.

Fig. 6. Offline calibration procedure (video available at http://www.youtube.com/watch?v=dEytfUgmcfA).

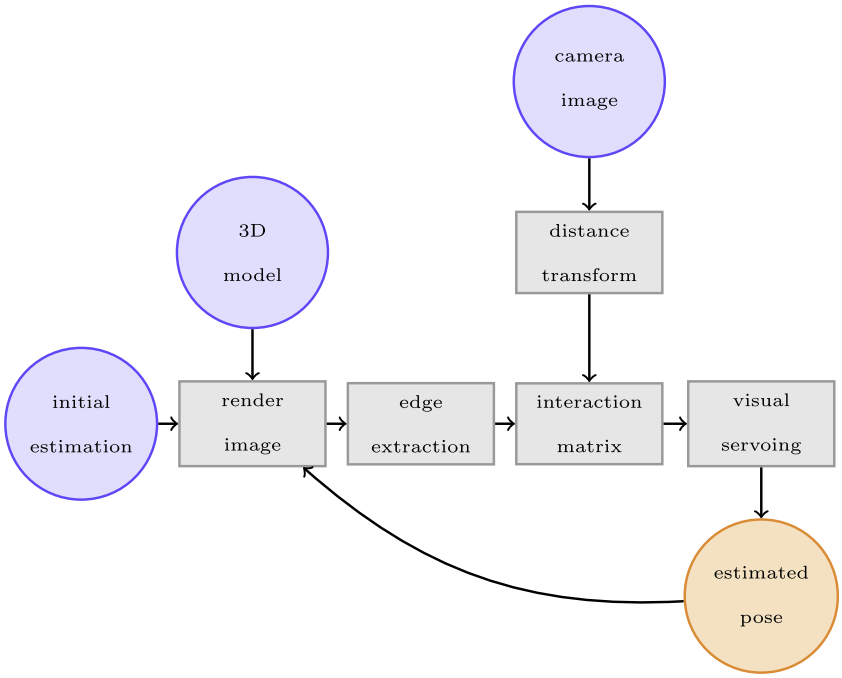


Fig. 7. Outline of the proposed model-based tracking system based on virtual visual servoing.

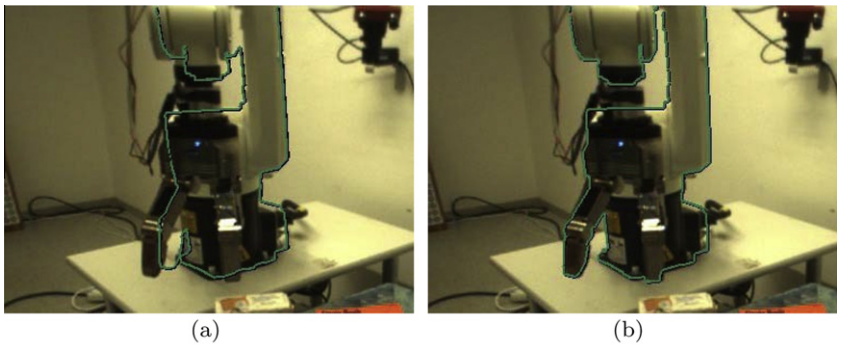


Fig. 8. Comparison of robot localization with (right) and without (left) the virtual visual servoing correction.

Based on the extracted features, we define an error vector

$$\boldsymbol{s}(t) = [d_1, d_2, \ldots, d_n]^{\mathrm{T}} \tag{16}$$

between the desired values for the features and the current ones. Based on $\boldsymbol{s}$, we can estimate the incremental change in pose in each iteration $\boldsymbol{e}(s)$ following the classical image-based servoing control loop. This process continues until the difference vector $\boldsymbol{s}$ is smaller than a certain threshold. Each of the steps is explained in more detail in the following subsections. Our current implementation and experimental evaluation is performed for the Kuka R850 arm and Schunk Dexterous hand, shown in Fig. 9.

4.5.1. Virtual image generation

As we mentioned before, we use a realistic 3D model of the robotic parts as input for our system. This adds the challenge of having to render this model at a high frame rate, since our system runs in real time, and several visual servoing iterations must be performed for every frame that we obtain from the cameras.

To render the image, we use a projection matrix $\boldsymbol{P}$, which corresponds to the internal parameters of the real camera, and a modelview matrix $\boldsymbol{M}$, which consists of a rotation matrix and a translation vector. The modelview matrix is then estimated in the visual servoing loop.

One of the most common CAD formats for objects such as robotic hands are Inventor files. There are a number of rendering engines which can deal with such models either natively or through the usage of plugins, such as Ogre [36] or OpenSceneGraph [37]. These rendering engines are designed and optimized to show the model appearance on the screen. However, we are interested in a very different task: obtaining the 3D points of the model surface directly in memory for further processing. This kind of rendering was slow, overcomplicated or even impossible in the rendering engines tested.

For this reason we developed a new scenegraph engine, specific for this application, which focused on rendering offscreen images at a high speed. It was developed directly over OpenGL. We can obtain about 1000 frames per second with this engine. At the moment the speed of the rendering engine does not represent a bottleneck to our system.

We render the image without any texture or lighting, since we are only interested in the external edges of the model. We also save the depth map produced by the rendering process, which will be useful later for the estimation of the jacobian.

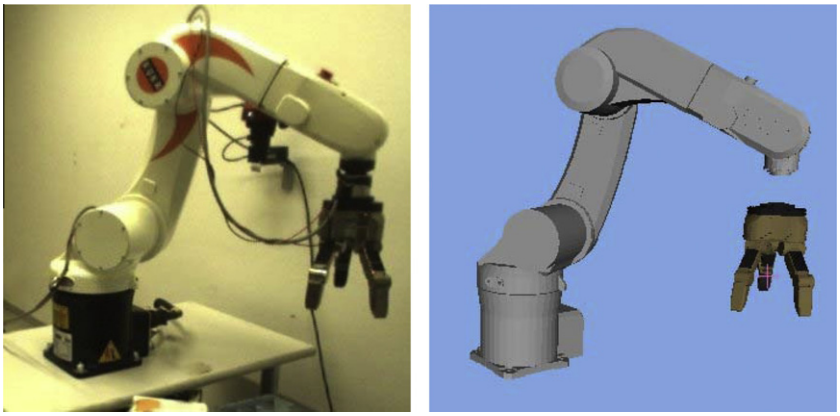


Fig. 9. The Kuka R850 arm and Schunk dexterous hand in real images and CAD models.

4.5.2. Features

The virtual and the real image of the robot parts ought to be compared in terms of visual features. These features should be fast to compute, because this comparison will be performed as many times per frame as iterations are needed by the virtual visual servoing for locating the robot. They should also be robust towards non-textured models.

Edge-based features fulfill these requirements. In particular chamfer distances [38] modified to include the alignment of edge orientations [39] are well suited for matching shapes in cluttered scenes, and are used in recent systems for object localization [40]. This feature is similar in spirit but more general than other ones used in the field of Virtual Visual Servoing. Comport et al. [41] computes distances between real points and virtual lines/ellipses instead of virtual points. Klose et al. [42],Drummond and Cipolla [43] search for real edges only in the perpendicular direction of the virtual edge.

The mathematic formulation of our features is the following. After performing a Canny edge detection on real image I_u and virtual image I_v we obtain a set of points lying on an edge $\mathcal{U},\mathcal{V}$ with their correspondent edge orientations $\mathcal{O}_\mathcal{U},\mathcal{O}_\mathcal{V}$ (from the horizontal and vertical Sobel filter). The set $\mathcal{V}$ has a typical cardinality of 1000. The speed of the servoing loop can be increased by randomly subsampling this set. We tested subsampling the set up to four times without noticeable decrease in the performance. The sets $\mathcal{U},\mathcal{V}$ are split into overlapping subsets $\mathcal{U}_i,\mathcal{V}_i$ according to their orientations:

$$o_u(\mathrm{mod}\,\pi) \in \left[\frac{2\pi i}{16}, \frac{2\pi(i+1)}{16}\right] \Rightarrow u \in \mathcal{U}_i \tag{17}$$

Then for each channel $\mathcal{U}_i$ we compute the distance transform [44] which will allow us to perform multiple distance computations for the same real set $\mathcal{U}$ in linear time with the number of points in each new set $\mathcal{V}$. Based on the distance transform we obtain our final distance vector $\boldsymbol{s} = \{d_{cham}(v,\mathcal{U})\} = [d_1, d_2 \cdots d_n]^\mathrm{T}$ composed by distance estimations for each edge point from our virtual image I_v.

$$d_{transf}(p) = \min_{u\in\mathcal{U}_i} \|p-u\|, \quad p \in I_u \tag{18}$$

$$d_{cham}(v,\mathcal{U}) = d_{transf}(v), \quad v \in \mathcal{V}_i, u \in \mathcal{U}_i \tag{19}$$

$$\boldsymbol{s} = \{d_{cham}(v,\mathcal{U})\}, \quad d_{cham}(v,\mathcal{U}) < \delta \tag{20}$$

The points $v \in \mathcal{V}$ whose distance is higher than an empirically estimated threshold δ are considered outliers and dropped from $\boldsymbol{s}$.

4.5.3. Pose correction

Once the features have been extracted, we can use a classical visual servoing approach to calculate the correction to the pose, [45]. There are two different approaches to vision-based control: *Position-based* control uses image data to extract a series of 3D features, and control is performed in the 3D Cartesian space. In *image-based* control, the image features are directly used to control the robot motion. In our case, since the features we are using are distances between edges in an image for which we have no depth information in the real image, we are using an image-based approach.

The basic idea behind visual servoing is to create an error vector which is the difference between the desired and measured values for a series of features, and then map this error directly to robot motion.

As discussed before, $\boldsymbol{s}(t)$ is a vector of feature values measured in the image, composed by distances between edges in the real and synthetic images I_u, I_v. Therefore $\dot{\boldsymbol{s}}(t)$ will be the rate of change of these distances with time.

The movement of the manipulator (in this case, the virtual manipulator) can be described by a translational velocity $\boldsymbol{T}(t) = [T_x(t), T_y(t), T_z(t)]^T$ and a rotational velocity $\boldsymbol{\Omega}(t) = [\omega_x(t), \omega_y(t), \omega_z(t)]^T$. Together, they form a velocity screw:

$$\dot{\boldsymbol{r}}(t) = [T_x, T_y, T_z, \omega_x, \omega_y, \omega_z]^\mathrm{T} \tag{21}$$

We can then define the image jacobian or interaction at a certain instant as $\boldsymbol{J}$ so that:

$$\dot{\boldsymbol{s}} = \boldsymbol{J}\dot{\boldsymbol{r}} \tag{22}$$

where

$$\boldsymbol{J} = \left[\frac{\partial \boldsymbol{s}}{\partial \boldsymbol{r}}\right] = \begin{bmatrix} \frac{\partial d_1}{\partial T_x} & \frac{\partial d_1}{\partial T_y} & \frac{\partial d_1}{\partial T_z} & \frac{\partial d_1}{\partial \omega_x} & \frac{\partial d_1}{\partial \omega_y} & \frac{\partial d_1}{\partial \omega_z} \\ \frac{\partial d_2}{\partial T_x} & \frac{\partial d_2}{\partial T_y} & \frac{\partial d_2}{\partial T_z} & \frac{\partial d_2}{\partial \omega_x} & \frac{\partial d_2}{\partial \omega_y} & \frac{\partial d_2}{\partial \omega_z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial d_n}{\partial T_x} & \frac{\partial d_n}{\partial T_y} & \frac{\partial d_n}{\partial T_z} & \frac{\partial d_n}{\partial \omega_x} & \frac{\partial d_n}{\partial \omega_y} & \frac{\partial d_n}{\partial \omega_z} \end{bmatrix} \tag{23}$$

which relates the motion of the (virtual) manipulator to the variation in the features. The method used to calculate the jacobian is described in detail below.

However, to be able to correct our pose estimation, we need the opposite: we need to compute $\dot{\boldsymbol{r}}(t)$ given $\dot{\boldsymbol{s}}(t)$.

When $\boldsymbol{J}$ is square and nonsingular, it is invertible, and then $\dot{\boldsymbol{r}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{s}}$. This is not generally the case, so we have to compute a least squares solution, which is given by

$$\dot{\boldsymbol{r}} = \boldsymbol{J}^{+}\dot{\boldsymbol{s}} \tag{24}$$

where $\boldsymbol{J}^+$ is the pseudoinverse of $\boldsymbol{J}$, which can be calculated as:

$$\boldsymbol{J}^{+} = (\boldsymbol{J}^\mathrm{T}\boldsymbol{J})^{-1}\boldsymbol{J}^\mathrm{T}. \tag{25}$$

The goal for our task is to have all the edges in our synthetic image match edges in the real image, so the target value for each of the features is 0. Then, we can define the error function as

$$\boldsymbol{e}(\boldsymbol{s}) = \mathbf{0} - \dot{\boldsymbol{s}} \tag{26}$$

which leads us to the simple proportional control law:

$$\dot{\boldsymbol{r}} = -K\boldsymbol{J}^{+}\boldsymbol{s} \tag{27}$$

where K is the gain parameter.

4.5.4. Estimation of the jacobian

To estimate the jacobian, we need to calculate the partial derivatives of the feature values with respect to each of the motion components. When features are the position of points or lines, it is possible to find an analytical solution for the derivatives.

In our case, the features in $\boldsymbol{s}$ are the distances from the edges of the synthetic image to the closest edge in the real image. Therefore, we numerically approximate the derivative by calculating how a small change in the relevant direction affects the value of the feature.

As we said before, while rendering the model we obtained a depth map. From this depth map, it is possible to obtain the 3D point corresponding to each of the edges. Each model point v in I_v is a projection of its corresponding 3D point in the model $\boldsymbol{v_m}$. The derivative of the distance described before $d_{cham}(v, \mathcal{U})$, can be calculated for a model point $\boldsymbol{v_m}$ with respect to T_x by applying a small displacement and projecting it into the image:

$$\frac{d_{cham}(\boldsymbol{PM}(\boldsymbol{v_m} + \epsilon\boldsymbol{x}), \mathcal{U}) - D(\boldsymbol{PMv_m}, \mathcal{U})}{\epsilon} \tag{28}$$

where ϵ is an arbitrary small number and $\boldsymbol{x}$ is a unitary vector in the x direction. $\boldsymbol{P}$ and $\boldsymbol{M}$ are the projection and modelview matrices, as defined in Section 4.5.1. A similar process is applied to each of the motion components.

4.5.5. Summary

In this section, we described how virtual visual servoing is used in our grasping pipeline to correct errors in the estimation of the pose of the robotic arm with respect to the cameras. The goal is to align the edges in the rendered model of the robot with the edges in the real image. This is achieved by iteratively applying incremental corrections $\dot{\boldsymbol{r}}$ to the motion components in a direction which minimizes the sum of the distances between the model edges and the real ones.

4.6. Object grasping

In this section we will combine the grasping point computed in Section 4.2 and the arm pose calculated in Section 4.5 in order to move the arm so that it can grasp the object.

After finding the grasping position as a point G^{C_0} in camera space, we move the head to a position where the whole arm is visible, while keeping the object in the field of view. Then, the object position (x,y) is found in this new viewpoint using the method in Section 4.3. We assume that the z coordinate did not change with the gaze shift. Therefore, we use the one calculated previously in G^{C_0}. The grasping point in camera space for the current viewpoint is then $G^{C_1} = [zx\ zy\ z]$.

After this, virtual visual servoing allows us to obtain the transformation $A^{A}_{C_1}$ that converts points from camera to arm space, so we can obtain the grasping point in arm space as $G^{A} = A^{A}_{C_1} G^{C_1}$. To account for small errors in the measurements, we do not move the arm there directly, but take it first to a position which is a few centimeters (15 cm in our experiments) above G^{A}.

Then, the final step is to bring the arm down so that the object lies between the fingers of the hand. We do this using a simple visual servoing loop. The movement to be performed is purely vertical. Therefore, we only use a single visual feature to control that degree of freedom: the vertical distance between the arm and the grasping point in the image which will be roughly aligned with the vertical axis of the arm. In each iteration, we measure the vertical distance d between the arm and the grasping point in the image, and use this as input for a simple proportional control law:

$$\dot{r}_y = -kd \tag{29}$$

where $\dot{r}_y$ is the vertical velocity of the arm and k is a gain factor.

5. Experiments

5.1. Robustness of virtual visual servoing

In order to evaluate the performance of the virtual visual servoing, we needed a setup where ground truth data was available, so that the error both for the input (edge detection and initial estimation) and the output (estimated pose) is known. Lacking such a setup, we decided to conduct the experiments using synthetic images as input. These images were generated using the same 3D model and rendering system that we use in the visual servoing loop.

The process used in these experiments was as follows:

- Generate a rendered image of the model at a known pose.
- Add some error in translation and rotation to that pose, and use this as the initial pose estimation.
- Run the virtual visual servoing loop with the rendered image as input. In some of the runs, noise was added to the detection of edges, to assess the robustness with respect to certain errors in the edge detection.
- After each iteration, check whether the method has reached a stable point (small correction) and the difference between the detected pose and the known pose is below a certain threshold. If this is the case, consider it a successful run and store the number of iterations needed.
- If the system does not converge to the known pose after a certain number of iterations, stop the process and count it as a failed run.

We ran three sets of experiments, each of them focusing on the following kind of input error:

- Translational error. We evaluated which range of errors in the translational component of the initial pose estimation allows the system to converge. To that effect, we added, in each run, a translational error of known magnitude and random direction.
- Rotational error. We also evaluated the range of errors in the rotational component of the initial pose estimation for which the system converges to the correct result. For each run, we added a rotational error of known magnitude and random direction.
- Error in edge detection. To simulate the effects of wrongly detected edges in the performance of the system, we added random errors to the edge detection of the input synthetic image. In each run, the detection of a number of pixels was shifted by a random amount. An example of the result can be seen in Fig. 10.

The performance of the VVS system is measured in (i) the number of runs that failed and (ii), for the runs that succeeded, the average number of iterations it took to do so. We plot these with respect to the magnitude of the input error. For each value of the input error, we ran 500 visual servoing loops. In each of these runs, the magnitude of the error was kept constant, but the direction (or the particular pixels that were affected in the case of edge detection) was chosen randomly. Also, the whole process was repeated for five different configurations of the joints of the arm.

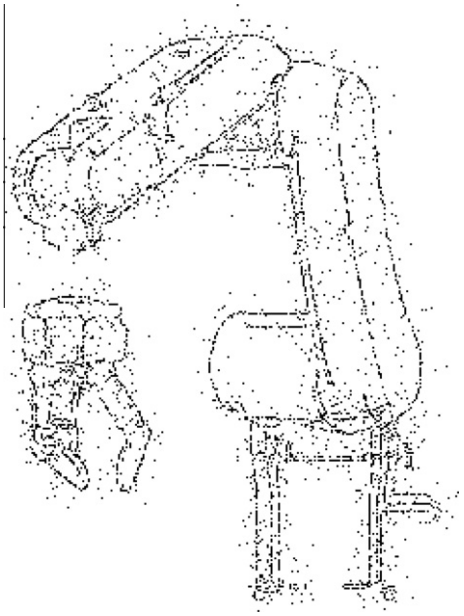


Fig. 10. Edge detection with artificially introduced error (in 30% of the pixels).

The rest of the parameters of the process were set as follows:

- The threshold for deciding that the algorithm had converged to the correct pose was 10 mm in translation and 0.5° in rotation. This threshold was empirically determined so that it guaranteed real convergence, and making it smaller would have caused stability issues.
- The number of iterations after which the run was considered a failure was set to 200.
- For the experiments that evaluate robustness with respect to edge detection, a translation error of 100 mm and a rotation error of 10° was used for the initial pose estimation.

Fig. 11 shows the result of increasing the translational component of the error in the estimated initial pose. As we can see, the failure rate is close to 0 for distances below 100 mm, and then starts increasing linearly. This is probably due to 100 mm being in the same order of magnitude as the distance between different edges of the robot which have the same orientation, so the system gets easily lost, trying to follow the wrong edges. We can also observe an increase in the iterations needed for reaching the result.

In Fig. 12 we can observe a similar behavior for the tolerance to errors in the rotational component of the estimated initial pose. Here, the threshold is around 10° and the reason is probably the same as before: this is the minimum rotation that bring edges to the position of other edges.

With respect to the error in edge detection, we can see in Fig. 13 that when less than 50% of the pixels are wrongly detected, the system performs almost as well as with no error, and after that the performance quickly degrades. It is also significant that for the runs that converge successfully, the number of iterations is almost independent of the errors in edge detection.

5.2. Qualitative experiments on the real system

For our experiments with the real robot, we set up a table-top scenario with two randomly placed objects, as can be seen in the last picture in Fig. 14. The head was initially looking towards the table, where it could see the objects and find them in the saliency map of the attention system. However, the arm was not fully visible.

We ran the system several times with this setup. In most of these runs, the virtual visual servoing loop did not converge because it could only see a small part of the arm. Therefore, we decided to run virtual visual servoing only after the object is found and after shifting the gaze away from it and towards the arm. With this change, even if the initial estimation for the pose of the arm was significantly different, the pose estimated through visual servoing quickly converged. In Fig. 14 we can see, outlined in green, the estimated pose for the arm and hand over the real image.

The segmentation of the object and detection of the grasping point worked well. When running virtual visual servoing with the full arm in view, the hand could be aligned with the object with high accuracy and grasping was performed successfully in most of the runs.

However, even if they did not affect the performance of the system, there are some problems that arise with the use of virtual visual servoing in real images. For example, as we can see in the upper-right picture of Fig. 14, the upper edge of the reconstructed outline does not match exactly with the upper edge of the arm. There are two main reasons for this. First, the illumination of the scene can lead to some edges disappearing, because of the low contrast. This is usually not a problem, because the correct matching of other edges will compensate for this. But in this case, there is a second problem: because of the orientation of the arm with respect to the camera, the outer silhouette has really few edges. As we can see in the picture below that, once the pose of the arm changes, the system can recover and show the correct pose again. As future work, we are considering the possibility of using not only the outer contour, but also try to match some of the internal edges of the model. While this can lead to a less robust system, since outer edges are more likely to appear as edges in the real image, it can increase performance where the number of edges is low.

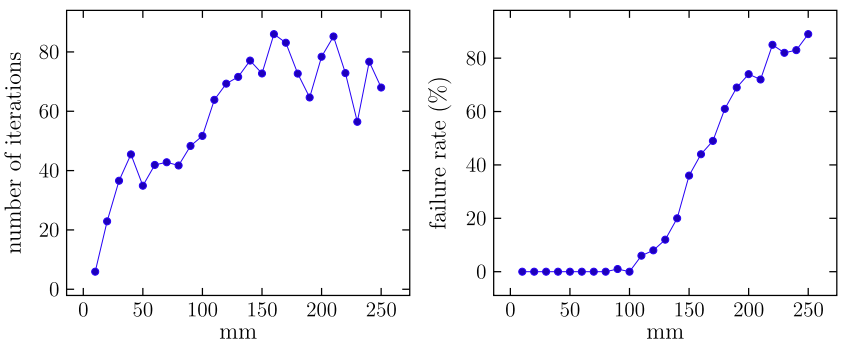


Fig. 11. Effects of translational error.

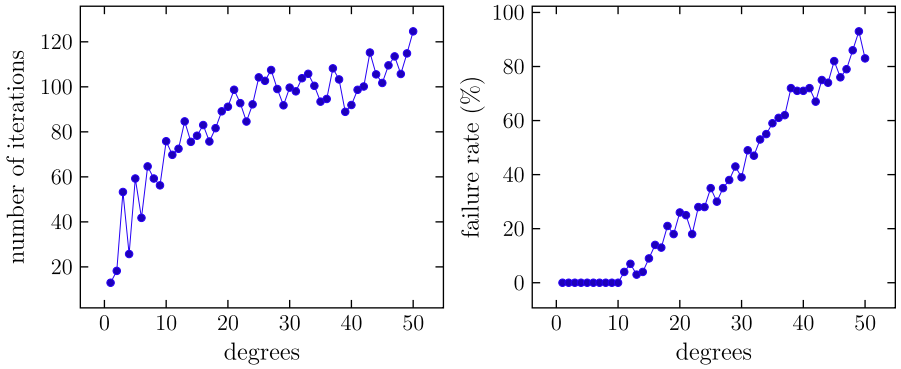


Fig. 12. Effects of rotational error.

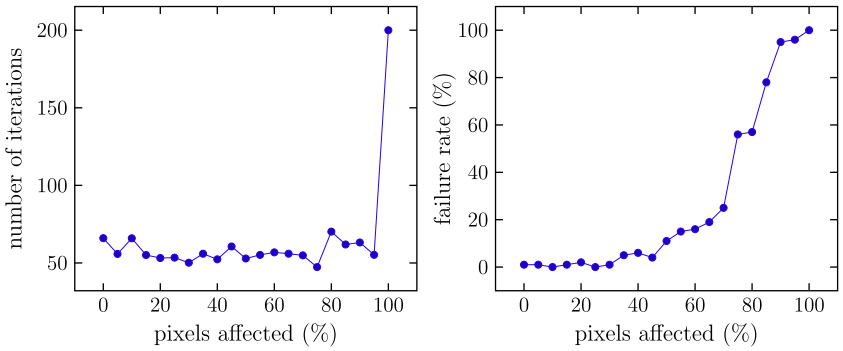


Fig. 13. Effects of errors in edge detection.

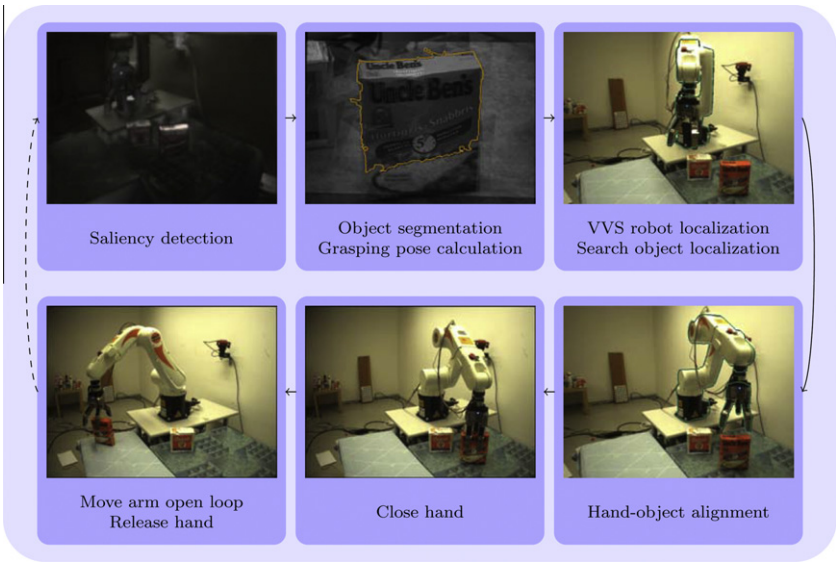


Fig. 14. Grasping system diagram. A video with the whole sequence can be found at http://www.youtube.com/watch?v=e-O3Y8_cgPw.

6. Conclusions

Grasping and manipulation of objects is a necessary capability of any robot performing realistic tasks in a natural environment. The solutions require a systems approach since the objects need to be detected and modelled prior to an agent acting upon them. Although many solutions for known objects have been proposed in the literature, dealing with unknown objects stands as an open problem.

Our research deals with this problem by integrating the areas of active vision and sensor based control where visual servoing methods represent important building blocks. In this paper, we have presented several ways in which these methods can add robustness and help minimize the errors inherent to any real system.

First, we showed how visual servoing can be used to automate the offline calibration process. The robot can, without human intervention, generate a pattern to collect data that can then be used to

calibrate the cameras and the transformations between the cameras and the arm.

Then, we demonstrated a virtual visual servoing approach to continuously correct the spatial relation between the arm and the cameras. Though in its current state the system works well and is useful in the context of the system, some ideas for future work arise from the experiments we performed. Using the edges as the only feature in the visual servoing loop works well with objects which have a complex outline, such as our arm. However, for other kinds of object, or even some viewpoints of the arm, having a wider range of features might help. These features may include adding textures to the models, or using 3D information obtained via stereo or structured light cameras.

Finally, the last step that brings the arm to the place where the object can actually be grasped also uses visual information to move the arm down to the position where the hand can be closed to grasp the object. There is also room for future improvements here, such as integrating the system with a more complex grasp planner, which could determine the optimal points where the fingers should contact the object. Then, visual servoing could be used to visually bring the fingers to the correct position.

Acknowledgments

This work is supported by the EU through the Project GRASP, IST-FP7-IP-215821, Swedish Foundation for Strategic Research and UKF-Unity through Knowledge Fund.

References

[1] Glover J, Rus D, Roy N. Probabilistic models of object geometry for grasp planning. In: IEEE international conference on robotics and automation, Pasadena, CA, USA; 2008.
[2] Huebner K, Welke K, Przybylski M, Vahrenkamp N, Asfour T, Kragic D, et al. Grasping known objects with humanoid robots: a box-based approach. In: International conference on advanced robotics; 2009. p. 1–6.
[3] Rasolzadeh B, Björkman M, Huebner K, Kragic D. An active vision system for detecting, fixating and manipulating objects in real world. Int J Robot Res.
[4] Azad P, Asfour T, Dillmann R. Stereo-based 6D object localization for grasping with humanoid robot systems. In: IEEE/RSJ int. conf. on intelligent robots and systems (IROS); 2007. p. 919–24.
[5] Bohg J, Kragic D. Learning grasping points with shape context. Robot Auton Syst 2010;58(4):362–77.
[6] Saxena A, Driemeyer J, Kearns J, Ng AY. Robotic grasping of novel objects. Neural Inform Process Syst 2007;19:1209–16.
[7] Speth J, Morales A, Sanz PJ. Vision-based grasp planning of 3D objects by extending 2D contour based algorithms. In: IEEE/RSJ international conference on intelligent robots and systems; 2008. p. 2240–5.
[8] Stark M, Lies P, Zillich M, Wyatt J, Schiele B. Functional object class detection based on learned affordance cues. In: 6th International conference on computer vision systems. LNAI, vol. 5008. Springer-Verlag; 2008. p. 435–44.
[9] Huebner K, Kragic D. Selection of robot pre-grasps using box-based shape approximation. In: IEEE/RSJ international conference on intelligent robots and systems; 2008. p. 1765–70.
[10] Richtsfeld M, Vincze M. Grasping of unknown objects from a table top. In: ECCV workshop on 'vision in action: efficient strategies for cognitive agents in complex environments', Marseille, France; 2008.
[11] Aarno D, Sommerfeld J, Kragic D, Pugeault N, Kalkan S, Wörgötter F, et al. Early reactive grasping with second order 3D feature relations. In: ICRA workshop: from features to actions; 2007. p. 319–25.
[12] Bergström N, Bohg J, Kragic D. Integration of visual cues for robotic grasping. In: Computer vision systems. Lecture notes in computer science, vol. 5815. Berlin/Heidelberg: Springer; 2009. p. 245–54.
[13] Kragic D, Christensen HI. Cue integration for visual servoing. IEEE Trans Robot Autom 2001;17(1):18–27.
[14] Asfour T, Welke K, Azad P, Ude A, Dillmann R. The Karlsruhe humanoid head. In: IEEE/RAS international conference on humanoid robots (Humanoids), Daejeon, Korea; 2008.
[15] KUKA, KR 5 sixx R850. <http://www.kuka-robotics.com> (last visited 12.10).
[16] SCHUNK, SDH. <http://www.schunk.com> (last visited 12.10).
[17] Gratal X, Bohg J, Björkman M, Kragic D. Scene representation and object grasping using active vision. In: IROS'10 workshop on defining and solving realistic perception problems in personal robotics; 2010.
[18] Felip J, Morales A. Robust sensor-based grasp primitive for a three-finger robot hand. In: Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems, IROS'09; 2009. p. 1811–6.
[19] Hsiao K, Chitta S, Ciocarlie M, Jones EG. Contact-reactive grasping of objects with partial shape information. In: IEEE/RSJ int. conf. on intelligent robots and systems (IROS), Taipei, Taiwan; 2010. p. 1228–35.
[20] Ude A, Omrcen D, Cheng G. Making object learning and recognition an active process. Int J Humanoid Robot 2008;5(2):267–86.
[21] Vahrenkamp N, Wieland S, Azad P, Gonzalez D, Asfour T, Dillmann R. Visual servoing for humanoid grasping and manipulation tasks. In: IEEE/RAS int. conf. on humanoid robots (Humanoids); 2008. p. 406–12.
[22] Ude A, Oztop E. Active 3-D vision on a humanoid head. In: Int. conf.on advanced robotics (ICAR), Munich, Germany; 2009.
[23] Welke K, Przybylski M, Asfour T, Dillmann R. Kinematic calibration for saccadic eye movements. Tech. Rep., Institute for Anthropomatics, Universität Karlsruhe; 2008.
[24] Pradeep V, Konolige K, Berger E, Calibrating a multi-arm multi-sensor robot: a bundle adjustment approach. In: International symposium on experimental robotics (ISER), New Delhi, India; 2010.
[25] Lepetit V, Pilet J, Fua P. Point matching as a classification problem for fast and robust object pose estimation. In: CVPR, II; 2004. p. 244–50.
[26] Drummond T, Cipolla R. Real-time visual tracking of complex structures. IEEE Trans PAMI 2002;24(7):932–46.
[27] Kyrki V, Kragic D. Integration of model-based and model-free cues for visual object tracking in 3D. In: IEEE international conference on robotics and automation, ICRA'05; 2005. p. 1566–72.
[28] Kragic D, Kyrki V. Initialization and system modeling in 3-D pose tracking. In: IEEE international conference on pattern recognition 2006. ICPR'06, Hong Kong; 2006. p. 643–6.
[29] Comport AI, Kragic D, Marchand E, Chaumette F. Robust real-time visual tracking: comparison, theoretical analysis and performance evaluation; 2005. p. 2841–6.
[30] Romero J, Kjellström H, Kragic D. Hands in action: real-time 3D reconstruction of hands in interaction with objects. In: ICRA, IEEE; 2010. p. 458–63.
[31] León B, Ulbrich S, Diankov R, Puche G, Przybylski M, Morales A, et al. OpenGRASP: a toolkit for robot grasping simulation. In: SIMPAR '10: proceedings of the 2nd international conference on simulation, modeling, and programming for autonomous robots; 2010.
[32] Björkman M, Kragic D. Active 3D scene segmentation and detection of unknown objects. In: IEEE international conference on robotics and automation; 2010a.
[33] Björkman M, Kragic D. Active 3D segmentation through fixation of previously unseen objects. In: British machine vision conference; 2010b.
[34] Itti L, Koch C. Computational modelling of visual attention. Nat Rev Neurosci 2001;2(3):194–203.
[35] Zhang Z. A flexible new technique for camera calibration. IEEE Trans Pattern Anal Mach Intell 2000;22:1330–4. ISSN:0162-8828.
[36] OGRE, <http://www.ogre3d.org/> (last visited 12.10).
[37] OpenSceneGraph, <http://www.openscenegraph.org/projects/osg> (last visited 12.10).
[38] Butt MA, Maragos P. Optimum design of chamfer distance transforms. IEEE Trans Image Process 1998;7(10):1477–84.
[39] Gavrila DM. Multi-feature hierarchical template matching using distance transforms. In: ICPR, vol. I; 1998. p. 439–44.
[40] Liu M, Tuzel O, Veeraraghavan A, Chellappa R. Fast directional chamfer matching. In: CVPR, IEEE; 2010. p. 1696–703.
[41] Comport AI, Marchand É, Pressigout M, Chaumette F. Real-time markerless tracking for augmented reality: the virtual visual servoing framework. IEEE Trans Vis Comput Graph 2006;12(4):615–28.
[42] Klose S, Wang J, Achtelik M, Panin G, Holzapfel F, Knoll A. Markerless, vision-assisted flight control of a quadrocopter. In: IROS; 2010.
[43] Drummond T, Cipolla R. Real-time visual tracking of complex structures. IEEE Trans Pattern Anal Mach Intell 2002;24(7):932–46.
[44] Borgefors G. Distance transformations in digital images, computer vision. Graph Image Process 1986;34:344–71.
[45] Hutchinson S, Hager G, Corke P. A tutorial on visual servo control. IEEE Trans Robot Autom 1996;12(5):651–70.

"Open Sesame!" Adaptive Force/Velocity Control for Opening Unknown Doors

Y. Karayiannidis, C. Smith, F. E. Viña, P. Ögren, and D. Kragic
Computer Vision and Active Perception Lab., Centre for Autonomous Systems,
School of Computer Science and Communication, Royal Institute of Technology (KTH)
SE-100 44 Stockholm, Sweden.
{yiankar|ccs|fevb|petter|dani}@kth.se

***Abstract*— The problem of door opening is fundamental for robots operating in domestic environments. Since these environments are generally less structured than industrial environments, several types of uncertainties associated with the dynamics and kinematics of a door must be dealt with to achieve successful opening. This paper proposes a method that can open doors without prior knowledge of the door kinematics. The proposed method can be implemented on a velocity controlled manipulator with force sensing capabilities at the end effector. The method consists of a velocity controller which uses force measurements and estimates of the radial direction based on adaptive estimates of the position of the door hinge. The control action is decomposed into an estimated radial and tangential direction following the concept of hybrid force/motion control. A force controller acting within the velocity controller regulates the radial force to a desired small value while the velocity controller ensures that the end effector of the robot moves with a desired tangential velocity leading to task completion. This paper also provides a proof that the adaptive estimates of the radial direction converge to the actual radial vector. The performance of the control scheme is demonstrated in both simulation and on a real robot.**

I. Introduction

A robot should be able to open doors as simple as saying "Open Sesame!". However, the task of opening a door — or a cupboard — in a domestic environment includes several types of uncertainty that disqualifies the use of motion control and trajectory planning that is effective for stiff industrial robots. The uncertainties in the manipulation of these kinematic mechanisms, e.g. doors and drawers, can be divided into (a) *dynamic uncertainties*, related to the dynamic model of the door or the drawer: door's inertia, dynamics of the hinge mechanism etc., and (b) *kinematic uncertainties* related to the kinematic model of the door or the drawer: type of joint that models the kinematic mechanism, which may be prismatic or revolute, size of the door, location of the hinge etc. This categorization has been used in several problems in robot control, like motion control [1] and force/motion control [2]. From a control perspective, the door opening problem can be regarded as a force/motion control problem in which the robot workspace can be divided into motion and force controlled subspaces according to the concept of hybrid force/motion control [3], [4].

In this work, we consider a general robotic setup with a velocity controlled manipulator equipped with a wrist force/torque sensor, and we propose an adaptive controller which can be easily implemented for dealing with the kinematic and dynamic uncertainties of doors. The proposed control scheme, which is inspired by the adaptive surface slope learning [5], does not require accurate identification of the motion constraint at each step of the door opening procedure, as opposed to existing solutions to the door opening problem (see Section II). It uses adaptive estimates of the radial direction which are constructed from estimates of the door's hinge position and converge during the procedure to the actual, dynamically changing, radial direction. It should be noted that the proposed method can also be applied to other types of manipulation under uncertain kinematic constraints. We have chosen the door opening problem as a concrete example, since it is well-studied and has well-defined constraints. The paper is organised as follows: In Section II we provide an overview of the related works for the door opening problem. Section III provides description of the kinematic and the dynamic model of the system and the problem formulation. The proposed solution and the corresponding stability analysis are given in Section IV followed by the simulation example of Section V. In Section VII the final outcome of this work is briefly discussed.

II. Related Work and our Contributions

Pioneering works on the door opening problem are the papers of [6] and [7]. In [6], experiments on door opening with an autonomous mobile manipulator were performed under the assumption of a known door model, using the combined motion of the manipulator and the mobile platform, while in [7], the estimation of the constraints describing the kinematics of the motion for the door opening problem is proposed.

The estimation technique of [7] is based on the observation that ideally, the motive force should be applied along the direction of the end-effector velocity. To overcome the problems of chattering due to measurement noise and ill-definedness of the normalization for slow end-effector motion, the authors propose estimation by spatial filtering, which may, however, cause lag and affect the system stability. The idea of using velocity measurements for estimating the direction of motion has inspired the recent work of [8] that uses a moving average filter in the velocity domain. An estimator is used to provide a velocity reference for an admittance controller. Ill-defined normalizations and estimation

lags are not dealt with. Estimation of the constraint using velocity measurements has also been used in [9], where velocity and impedance control have been used along the tangent and the radial axis of the door opening trajectory respectively.

Several position-based estimation techniques have also been proposed. In [10], the recorded motion of the end-effector is used in a least-squares approximation algorithm to estimate the center and the radius of the motion arc, and a compliant controller is used to cancel the effects of the high forces exerted due to inaccurate trajectory planning.

An optimization algorithm using the position of the end-effector was used in [11], [12]. The algorithm produces estimates of the radius and the center of the door and, subsequently of the control directions. The velocity reference is composed of a feedforward estimated tangential velocity and radial force feedback along. An equilibrium point control law enables a viscoelastic behavior of the system around an equilibrium position.

In [13], an inverse Jacobian velocity control law with feedback of the force error following the Task Space Formalism [14] is considered. In order to obtain the natural decomposition of the task, which is essential within this framework, the authors propose to combine several sensor modalities so that robust estimation is established. In [13], the estimation is based on the end-effector trajectory, to align the task frame with the tangent of the hand trajectory.

Other works estimate the geometry of the door off-line, prior to manipulation. In [15], a multi-fingered hand with tactile sensors grasping the handle is used, and the geometry of the door is estimated by observing the positions of the fingertips position while slightly and slowly pulling and pushing the door in position control. In a subsequent step, the desired trajectory is derived from the estimation procedure, and is used in a position controller. In [16], a probabilistic framework in order to learn the kinematic model of articulated objects in terms of object's parts connectivity, degrees of freedom of the objects and kinematic constraints is proposed. The learning procedure requires a set of motion observations of the objects, e.g. doors. The estimates are generated in an off-line manner and can feed force/position cartesian controllers [17]. Probabilistic methods — particle filters and extended Kalman filters — for mobile manipulation have also been applied for opening doors under uncertainty, in [18]. The authors use an a priori defined detailed model of the door, and simultaneously estimate position of the robot and the angle of the door.

Another part of the literature on the door opening problem exploits advanced hardware capabilities to accomplish the manipulation task. In [19], a combination of tactile-sensor and force-torque sensor is used to control the position and the orientation of the end-effector with respect to the handle. In [20], a specific hardware configuration with clutches that disengage selected robot motors from the corresponding actuating joints and hence enable passive rotation of these joints is used. Since no force sensing is present, a magnetic end-effector was used which cannot always provide the appropriate force for keeping the grasp of the handle fixed. In, [21] the authors exploited the extensive abilities of the hardware, and used joint torque measurements to realize Cartesian impedance control of the DLR lightweight robot II in order to open a door. In [22], the authors present experiments using a force/torque sensor on a custom lightweight robot to define the desired trajectory for a door opening task. In [23], a method for door opening is proposed that uses an impulsive force exerted by the robot to the door which is assumed to be a swinging door. A specific dynamic model for the door dynamics is used to calculate the initial angular velocity which is required for a specific change of the door angle, and implemented on the humaoid robot HRP-2.

In this paper, we propose a method that differs from the existing work by simultaneously providing the following benefits:

- **On-line performance**. Our method does not require preparatory measurements or detailed modelling, or an off-line estimation phase before the manipulation thask. Instead, our method allows the manipulator to open a previously unknown door as smooth as a known one.
- **Moderate hardware requirements**. Our method can be implemented on any manipulator that can be velocity controlled in either joint space or Cartesian space, with force measurements at the end effector or wrist.
- **Provable performance under uncertainty**. The proposed method explicitly includes the uncertain estimates in the controller, and we provide proof of convergence of the estimates, and of the stability of the proposed method even with initially large errors in the estimates.

III. System and Problem Description

In this section we define the problem of door opening under uncertainty.

A. Notation and Preliminaries

We introduce the following notation:

- Bold roman small letters denote vectors while bold roman capital letters denote matrices.
- The generalized position of a moving frame $\{i\}$ with respect to a inertial frame $\{B\}$ (located usually at the robots base) is described by a position vector $\mathbf{p}_i \in \mathbb{R}^m$ and a rotation matrix $\mathbf{R}_i \in SO(m)$ where $m = 2$ or 3 for the planar and spatial case respectively.
- We consider also the following normalization and orthogonalization operators:

$$\underline{\mathbf{z}} = \frac{\mathbf{z}}{\|\mathbf{z}\|} \tag{1}$$

$$\mathbf{s}(\mathbf{z}) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \mathbf{z} \tag{2}$$

with $\mathbf{z}$ being any non-trivial two dimensional vector. Notice that in case of $\mathbf{z} = \mathbf{z}(t)$ the derivative of $\underline{\mathbf{z}}$ is calculated as follows:

$$\dot{\underline{\mathbf{z}}} = \|\mathbf{z}\|^{-1} \mathbf{s}(\underline{\mathbf{z}}) \mathbf{s}(\underline{\mathbf{z}})^\top \dot{\mathbf{z}}. \tag{3}$$

- We denote with $\mathcal{I}(z)$ the integral of some scalar function of time $z(t) \in \mathbb{R}$ over the time variable t, i.e:

$$\mathcal{I}(z) = \int_0^t z(\tau) d\tau \tag{4}$$

B. Kinematic model of robot door opening

We consider a setting of a robot manipulator in which its end-effector has achieved a fixed grasp of the handle of a kinematic mechanism e.g. a door in a domestic environment. We use the term fixed grasp to denote that there is no relative translational velocity between the handle and the end-effector but we place no constraints on the relative rotation of the end-effector around the handle. We consider also that the motion of the handle is inherently planar which consequently implies a planar problem definition.

Let $\{e\}$ and $\{o\}$ be the end-effector and the door frame respectively (Fig. 1); the door frame $\{o\}$ is attached at the hinge which in our case is the center of door-mechanism rotation. The radial direction vector $\mathbf{r}$ is defined as the relative position of the aforementioned frames:

$$\mathbf{r} \triangleq \mathbf{p}_o - \mathbf{p}_e \tag{5}$$

By expressing $\mathbf{r}$ with respect to the door frame and differentiating the resultant equation we get:

$$\dot{\mathbf{R}}_o {}^o\mathbf{r} + \mathbf{R}_o {}^o\dot{\mathbf{r}} = \dot{\mathbf{p}}_o - \dot{\mathbf{p}}_e \tag{6}$$

By substituting ${}^o\dot{\mathbf{r}} = \dot{\mathbf{p}}_o = 0$ as well as $\dot{\mathbf{R}}_o = \omega \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \mathbf{R}_o$ with ω being the rotational velocity of the door, we get:

$$\dot{\mathbf{p}}_e = -\mathbf{s}(\mathbf{r})\omega \tag{7}$$

which describes the first-order differential kinematics of the door opening problem in case of a revolute hinge. Notice that the end-effector velocity along the radial direction of the motion is zero, i.e:

$$\mathbf{r}^\top \dot{\mathbf{p}}_e = 0 \tag{8}$$

The latter can be regarded as the constraint on the robot end-effector velocity.

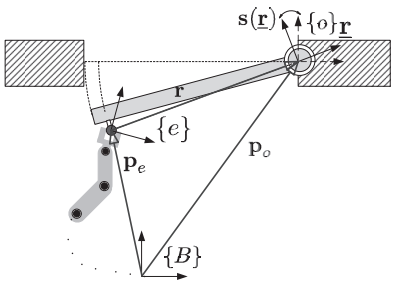


Fig. 1: Kinematics of the door opening

C. Robot kinematic model

In case of velocity controlled manipulators, the robot joint velocity is controlled directly by the reference velocity $\mathbf{v}_{\text{ref}}$. In particular, the reference velocity $\mathbf{v}_{\text{ref}}$ can be considered as a kinematic controller which is mapped to the joint space in order to be applied at the joint velocity level as follows:

$$\dot{\mathbf{q}} = \mathbf{J}^+(\mathbf{q})\mathbf{v}_{\text{ref}} \tag{9}$$

with $\mathbf{q},\ \dot{\mathbf{q}} \in \mathbb{R}^n$ being the joint positions and velocities and $\mathbf{J}(\mathbf{q})^+ = \mathbf{J}(\mathbf{q})^\top \left[\mathbf{J}(\mathbf{q})\mathbf{J}(\mathbf{q})^\top\right]^{-1}$ being the pseudo-inverse of the manipulator Jacobian $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{2\times n}$ which relates the joint velocities $\dot{\mathbf{q}}$ to the end-effector velocities $\dot{\mathbf{p}}_e$; without loss of generality we have here considered only the translational end-effector velocity $\dot{\mathbf{p}}_e \in \mathbb{R}^2$ and the associated Jacobian. If we consider the typical Euler-Lagrange robot dynamic model, the velocity error at the joint level drive the torque (current) controller $\mathbf{u}(t)$. If we assume a high frequency current control loop with external forces' compensators and weak inertial dynamics, then the kinematic model is valid.

D. Control Objective

The objective is to control the motion of the robot to achieve a smooth interaction with an external kinematic mechanism such as a door, and manipulate it in order to achieve a high level command such as "Open Sesame!". In applications which take place in a dynamic unstructured environments such as a domestic environment, it is difficult to accurately identify the position of the hinges and the associated dynamics. Hence, it is difficult to design a priori the desired velocity within the constraints imposed by the kinematic mechanism. The execution of a trajectory which is inconsistent with system constraints gives rise to high interaction forces along the constraint direction which may be harmful for both the manipulated mechanism and the robot.

Let f_{rd} and v_d be the desired radial force and desired tangent velocity magnitudes respectively. If we define the force along the radial direction as $f_r = \underline{\mathbf{r}}^\top \mathbf{F}$ with $\mathbf{F} \in \mathbb{R}^2$ being the total interaction force, the control objective can be formulated as follows: $f_r \to f_{rd}$ and $\dot{\mathbf{p}}_e \to \underline{\mathbf{r}} v_d$. These objectives have to be achieved without knowing accurately the $\underline{\mathbf{r}}$ direction which subsequently implies that there are uncertainties in the control variables f_r and $\mathbf{s}(\underline{\mathbf{r}})v_d$. From a high level perspective, we consider that the door opening task is accomplished when the observed end-effector trajectory, which coincides with the handle trajectory, enable the robot to perform the subsequent task which can be for example "get an object" or "pass through the door". Thus the command to halt the door opening procedure is given externally based on the observations of the rotation angle ϑ.

IV. Control Design

A. Incorporating Force Feedback in the Velocity Reference

Let us first define an estimated radial direction $\hat{\mathbf{r}}(t)$ based on appropriately designed adaptive estimates of the center of

rotation $\hat{\mathbf{p}}_o(t)$:

$$\hat{\mathbf{r}}(t) = \hat{\mathbf{p}}_o(t) - \mathbf{p}_e \tag{10}$$

For notation convenience we will drop out the argument of t from $\hat{\mathbf{r}}(t)$ and $\hat{\mathbf{p}}_o(t)$. We will use the estimated radial direction (10) considering that $\|\hat{\mathbf{r}}(t)\| \neq 0,\ \forall t$ in order to introduce a reference velocity vector $\mathbf{v}_{\text{ref}}$ for controlling the end-effector velocity:

$$\mathbf{v}_{\text{ref}} = \mathbf{s}(\underline{\hat{\mathbf{r}}})v_d - \alpha\underline{\hat{\mathbf{r}}}v_f \tag{11}$$

with α being a positive control gain acting on the force feedback term v_f which has been incorporated in the reference velocity.

We can now introduce the velocity error:

$$\tilde{\mathbf{v}} \triangleq \mathbf{v} - \mathbf{v}_{\text{ref}} \tag{12}$$

where $\mathbf{v} \triangleq \dot{\mathbf{p}}_e$ can be decomposed along $\underline{\hat{\mathbf{r}}}$ and $\mathbf{s}(\underline{\hat{\mathbf{r}}})$ and subsequently expressed with respect to the parameter estimation error $\tilde{\mathbf{p}}_o = \tilde{\mathbf{r}} = \mathbf{p}_o - \hat{\mathbf{p}}_o$ by adding $-\|\hat{\mathbf{r}}\|^{-1}\underline{\hat{\mathbf{r}}}\mathbf{r}^\top\mathbf{v}$ as follows:

$$\mathbf{v} = \mathbf{s}(\underline{\hat{\mathbf{r}}})\mathbf{s}(\underline{\hat{\mathbf{r}}})^\top\mathbf{v} - \|\hat{\mathbf{r}}\|^{-1}\underline{\hat{\mathbf{r}}}\tilde{\mathbf{p}}_o^\top\mathbf{v} \tag{13}$$

Substituting (13) and (11) in (12) we can obtain the following decomposition of the velocity error along the estimated radial direction $\underline{\hat{\mathbf{r}}}$ and the estimated direction of motion $\mathbf{s}(\underline{\hat{\mathbf{r}}})$:

$$\tilde{\mathbf{v}} = \hat{\mathbf{R}}_o \begin{bmatrix} -\|\hat{\mathbf{r}}\|^{-1}\tilde{\mathbf{p}}_o^\top\mathbf{v} + \alpha v_f \\ \mathbf{s}(\underline{\hat{\mathbf{r}}})^\top\mathbf{v} - v_d \end{bmatrix} \tag{14}$$

where $\hat{\mathbf{R}}_o \triangleq \begin{bmatrix} \underline{\hat{\mathbf{r}}} & \mathbf{s}(\underline{\hat{\mathbf{r}}}) \end{bmatrix}$.

In the next step, we are going to design the force feedback v_f employed in the reference velocity $\mathbf{v}_{\text{ref}}$. The force feedback term v_f is derived from the magnitude of the measured force components projected along the estimated radial direction:

$$\hat{f}_r = \underline{\hat{\mathbf{r}}}^\top\mathbf{F} \tag{15}$$

the corresponding force error:

$$\Delta\hat{f}_r = \hat{f}_r - f_{rd} \tag{16}$$

as well as the corresponding force error integral $\mathcal{I}(\Delta\hat{f}_r)$. In particular, for velocity controlled robotic manipulators, we propose a PI control loop of the estimated radial force error $\Delta\hat{f}_r$:

$$v_f = \Delta\hat{f}_r + \beta\mathcal{I}(\Delta\hat{f}_r) \tag{17}$$

with β being a positive control gain. By projecting $\tilde{\mathbf{v}} = 0$ along $\underline{\mathbf{r}}$ we can calculate $\hat{f}_r$ as a Lagrange multiplier associated with the constraint (6) for the system (9):

$$\hat{f}_r = f_{rd} - \beta\mathcal{I}(\Delta\hat{f}_r) + \frac{v_d\underline{\mathbf{r}}^\top\mathbf{s}(\underline{\hat{\mathbf{r}}})}{\alpha\underline{\mathbf{r}}^\top\underline{\hat{\mathbf{r}}}}. \tag{18}$$

Equation (18) is well defined for $\mathbf{r}^\top\hat{\mathbf{r}}(t) > 0$. Equation (18) is consistent to (15) in case of rigid contacts and fixed grasps.

Remark 1: For torque controlled robotic manipulators, the derivative of reference velocity also known as reference acceleration is required in the implementation. In order to avoid the differentiation of the force measurements in case of torque controlled manipulators, the force feedback part of the reference velocity should be designed using only the integral of the estimated radial force error.

B. Update Law Design

The update law for the vector $\hat{\mathbf{p}}_o$ is designed via a passivity-based approach, by defining the output of the system as follows:

$$y_f = \alpha_f\Delta\hat{f}_r + \alpha_I\mathcal{I}(\Delta\hat{f}_r) \tag{19}$$

with α_f and α_I being positive constants. Taking the inner product of $\tilde{\mathbf{v}}$ (14) with $\underline{\hat{\mathbf{r}}}y_f$ (19) we obtain:

$$\begin{aligned} y_f\underline{\hat{\mathbf{r}}}^\top\tilde{\mathbf{v}} &= y_f(-\|\hat{\mathbf{r}}\|^{-1}\tilde{\mathbf{p}}_o^\top\mathbf{v} + v_f) \\ &= -\|\hat{\mathbf{r}}\|^{-1}y_f\mathbf{v}^\top\tilde{\mathbf{p}}_o + c_1\Delta\hat{f}_r^2 \\ &\quad + c_2\mathcal{I}(\Delta\hat{f}_r)^2 + c_3\frac{d}{dt}\left[\mathcal{I}(\Delta\hat{f}_r)^2\right] \end{aligned} \tag{20}$$

where:

$$c_1 = \alpha\alpha_f, \quad c_2 = \alpha\alpha_I\beta, \quad c_3 = \frac{\alpha(\alpha_f\beta + \alpha_I)}{2} \tag{21}$$

Next, we design the update law $\dot{\hat{\mathbf{p}}}_o \triangleq -\dot{\tilde{\mathbf{p}}}_o$ as follows:

$$\dot{\hat{\mathbf{p}}}_o = \mathcal{P}\{\gamma\|\hat{\mathbf{r}}\|^{-1}y_f\mathbf{v}\} \tag{22}$$

Notice that $\mathcal{P}$ is an appropriately designed projection operator [24] with respect to a convex set of the estimates $\hat{\mathbf{p}}_o$ around $\mathbf{p}_o$ (Fig. 2) in which the following properties hold: i) $\|\hat{\mathbf{r}}\| \neq 0, \quad \forall t$, in order to enable the implementation of the reference velocity and calculate estimated radial force and ii) $\mathbf{r}^\top\hat{\mathbf{r}} > 0$; which is required for the system's stability. It

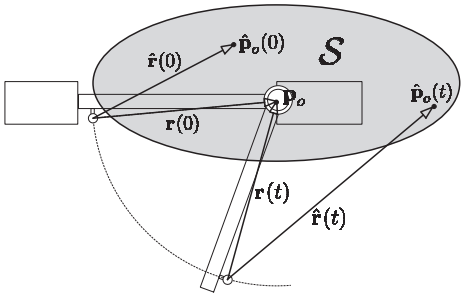


Fig. 2: Convex set $\mathcal{S}$ for the projection operator $\mathcal{P}$

is clear that the update law (22) gives rise to the potential owing to estimation error i.e. $\frac{1}{2\gamma}\tilde{\mathbf{p}}_o^\top\tilde{\mathbf{p}}_o$ and allow us to use the following function $V\left(\mathcal{I}(\Delta\hat{f}_r), \tilde{\mathbf{p}}_o\right)$ in order to prove Theorem 1 for velocity controlled manipulators. In particular $V\left(\mathcal{I}(\Delta\hat{f}_r), \tilde{\mathbf{p}}_o\right)$ is given by:

$$V\left(\mathcal{I}(\Delta\hat{f}_r), \tilde{\mathbf{p}}_o\right) = c_3\mathcal{I}(\Delta\hat{f}_r)^2 + \frac{1}{2\gamma}\tilde{\mathbf{p}}_o^\top\tilde{\mathbf{p}}_o \tag{23}$$

and is positive-definite with respect to $\mathcal{I}(\Delta\hat{f}_r)$, $\tilde{\mathbf{p}}_o$ and Theorem 1 is stated below:

Theorem 1: The kinematic controller $\mathbf{v}_{\text{ref}}$ (11) with the update law (22) applied to the system (9) achieves the following objectives: $\underline{\hat{\mathbf{r}}} \to \underline{\mathbf{r}}$, $\mathbf{v} \to \mathbf{s}(\underline{\mathbf{r}})v_d$, $\mathcal{I}(\Delta f_r) \to 0$ and

$f_r \rightarrow f_{rd}$, which are equivalent with the control objective of smooth door opening stated in Section III-D.

Proof: Substituting (11) in (9) and multiplying by $\mathbf{J}(\mathbf{q})$, implies $\tilde{\mathbf{v}} = 0$. Differentiating $V\left(\mathcal{I}(\Delta\hat{f}_r), \tilde{\mathbf{p}}_o\right)$ with respect to time and in turn substituting $\tilde{\mathbf{v}} = 0$ and (22) we get: $\dot{V} = -c_1\Delta\hat{f}_r^2 - c_2\mathcal{I}(\Delta\hat{f}_r)^2$; notice that $\dot{V}$ has extra negative terms when the estimates reach the bound of the convex set and the projection operator applies and thus the stability properties of the system are not affected. Hence, $\mathcal{I}(\Delta\hat{f}_r)$, $\tilde{\mathbf{p}}_o$ are bounded and consequently we can prove the boundedness of the following variables: (a) $\hat{f}_r$ is bounded, given the use of projection operator in (18), (b) $\mathbf{v}_{\text{ref}}$ is bounded, (c) $\dot{\mathbf{q}}$ is bounded, given the assumption of a non-singular manipulator in (9), (d) $\dot{\hat{\mathbf{p}}}_o$ is bounded, given (22) and the boundedness of $\mathbf{v}$.

The boundedness of the aforementioned variables implies that $\dot{\hat{f}}_r$ and subsequently $\ddot{V} = -2\Delta\hat{f}_r[c_1\dot{\hat{f}}_r + c_2\mathcal{I}(\Delta\hat{f}_r)]$ are bounded and thus Barbalat's Lemma implies $\dot{V} \rightarrow 0$ and in turn $\mathcal{I}(\Delta\hat{f}_r),\ \Delta\hat{f}_r \rightarrow 0$. Substituting the convergence results in (9) and (18) we get $\mathbf{v} \rightarrow \mathbf{s}(\underline{\hat{\mathbf{r}}})v_d$ and $\underline{\hat{\mathbf{r}}}^\top\mathbf{s}(\underline{\mathbf{r}}) \rightarrow 0$ for $\lim_{t\rightarrow\infty}|v_d| \neq 0$ (or for a v_d satisfying the persistent excitation condition) respectively; the latter implies $\underline{\hat{\mathbf{r}}} \rightarrow \underline{\mathbf{r}}$. Since the estimated direction of the constraint is identified we get: $\mathbf{v} \rightarrow \mathbf{s}(\underline{\mathbf{r}})v_d$, $\mathcal{I}(\Delta f_r) \rightarrow 0$ and $f_r \rightarrow f_{rd}$. □

C. Summary and Discussion

The proposed technique is based on a reference velocity (11) which is decomposed to a feedforward velocity on the estimated direction of motion and a PI force control loop on the estimated constrained direction. The estimated direction is obtained on-line using the update law (22) and the definition of the radial estimate (10). The use of (22) and (10) within a typical velocity reference like (11) enables the proof of the overall scheme stability as well as the proof that the estimates will converge to the actual values, driving the velocity and the radial force to their desired values.

Note that the proposed control scheme can be easily implemented using a very common robotic setup with a velocity-controlled robotic manipulator with a force/torque sensor in the end-effector frame. It is also clear that the proposed method is inherently on-line and explicitly includes the uncertain estimates in the controller, as opposed to the state of the for door opening (as described in Section II), which assumes that the estimate obtained in each step is approximately equal to the actual value. The proposed method can be also combined with off-line door kinematic estimation; in this case the off-line estimates can be used as the initial estimates of the estimator (22). However, our scheme is proven to work satisfactorily even in the case of large estimation errors, where off-line methods fail. Last but not least, the proposed method can be also be applied to other types of robot manipulation under kinematic uncertainties. We have chosen here the door opening problem since it is very challenging, but can be described in terms of concrete motion constraints.

V. Experimental Evaluation Using Simulation

We consider a 2 DoF robot manipulator (Fig. 3) which is modeled after one of the two 7 DoF arms of a semi-anthropomorphic robot at CAS/KTH, with only 2 DoFs being actuated while the remaining DoFs are mechanically braked. In particular, we consider that the second and fifth joints are actuated (red cylinders in Fig. 3) and simulate the case where this 2 DoFs planar manipulator can open a door through a fix-grasp of the cupboard handle. The DH parameters of the 7 DoF arm are shown in Table I. Regarding the kinematic parameters of the door, the center of rotation is $\mathbf{p}_o = [-0.8462\ 0.3722]^\top$ (m) in the robot world frame, while the length of the door (from hinge to handle) is approximately 0.51 m. In the simulation, the motion along the radial direction is governed by a stiff viscoelastic model while viscous friction is considered for the rotational motion.

Frame	α	a	θ	d
Base	$0°$	0.274	$90°$	0
Base	$90°$	0	$0°$	0
1	0	0	θ_1	0
2	$-90°$	0	$\theta_2 + 180°$	0
3	$-90°$	0	$\theta_3 + 180°$	0.313
4	$-90°$	0	$\theta_4 + 180°$	0
5	$-90°$	0	$\theta_5 + 180°$	0.2665
6	$-90°$	0	$\theta_6 + 180°$	0
7	$-90°$	0	$\theta_7 + 180°$	0
Tool	$0°$	0	$0°$	0.42

TABLE I: DH parameters of the 7-DOF arm (Craig's convention).

We set the initial estimate of the center of rotation $\hat{\mathbf{p}}_{o1}(0) = [-0.8462\ 0.7722]^\top$ (m) in the robot world frame which corresponds to the actual center of rotation misplaced for 40 cm along the x-axis; the initial uncertainty angle formed between the actual and the estimated radial direction is approximately 50 deg in this case which is extremely large. The controller objectives are set as follows: $v_d = 0.25$ m/s and $f_{rd} = 2$ N. The controller gains are chosen as follows: $a_f = 1$, $a_I = 0.8$, $\alpha = 0.6$, $\beta = 0.5$, $\gamma = 4$. Fig. 4 shows the top view of the manipulator while opening the door while Fig. 5 depicts the force error $\Delta f_r = f_r - f_{rd}$ response as well as the estimation error variable $e_r = 1 - \underline{\hat{\mathbf{r}}}^\top\underline{\mathbf{r}}$ response. Fig. 6 shows the velocity commands expressed in the joint space. Notice that very fast convergence in approximately 0.5 s is achieved (Fig. 5), but the demands of velocity (Fig. 6) is extremely high as compared to the maximum joint velocities in our experimental setup. (0.7 rad/s)

In the following simulations, we used the gains and considered the scenarios of Section VI (Experimental evaluation using Robot Platform). In the first scenario we consider the initial estimate of the center of rotation $\hat{\mathbf{p}}_{o1}(0)$ used before, by reducing the desired velocity to $v_d = 0.05$ m/s, while in the second scenario we consider that the center of rotation is misplaced for 5 cm along the x-axis of the robot world frame, i.e. $\hat{\mathbf{p}}_{o2}(0) = [-0.8462\ 0.4222]^\top$ (m) by setting the desired velocity to a higher value $v_d = 0.1$ m/s. The force control objective is set $f_{rd} = 2$ N and the controller gains are chosen as follows: $a_f = 0.1$, $a_I = 0.05$, $\alpha = 0.001$, $\beta = 0.1$, $\gamma =$

0.5, for both cases of initial uncertainty. In the latter case, the initial uncertainty angle formed between the actual and the estimated radial direction is approximately 5 deg. Simulation results (force and estimation errors' responses) are shown in Fig. 7 and 8 for the case of higher and lower uncertainty respectively. Notice that the estimation error converges to zero in approximately 1 s while the convergence of the force error is slower. Notice also that the overshoot in the force error is much bigger in case of higher uncertainty (Fig. 7), but as the controller finally tracks the actual direction it slowly vanishes.

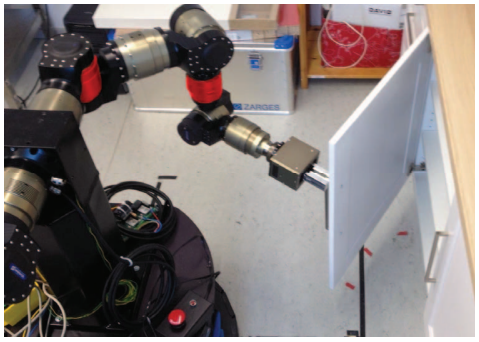


Fig. 3: 7 DoF arm of a semi-anthropomorphic robot at CAS/KTH; the figure has been digitally enhanced to mark the joints used for experimental evaluation with red hue

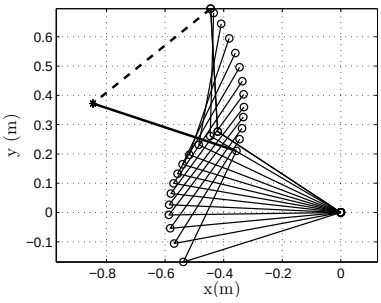


Fig. 4: Manipulator trace and door's initial (dashed bold line) and final(solid bold line) position

VI. Experimental Evaluation using Robot Platform

The performance was also evaluated on the real robot system. We consider the robot and door kinematic setup used in Section V. The arm is constructed from Schunk rotary modules, that can be sent velocity commands over a CAN bus. The modules incorporate an internal PID controller that keeps the set velocity, and return angle measurements. In this setup, the modules are sent updated velocity commands at 400 Hz. Angle measurements are read at the same frequency. The arm has an ATI Mini45 6 DoF force/torque sensor mounted at the wrist. The forces are also read at 400 Hz in this experiment. The force readings display white measurement noise with a magnitude of approximately 0.2 N, apart from any process noise that may be present in the mechanical

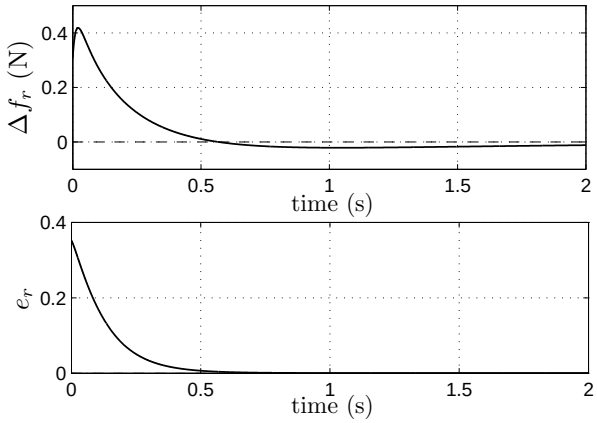


Fig. 5: Radial force error (Upper plot) and Estimation Error (Lower plot) responses - Simulation for $\hat{\mathbf{p}}_{o1}(0)$, $v_d = 0.25$ m/s

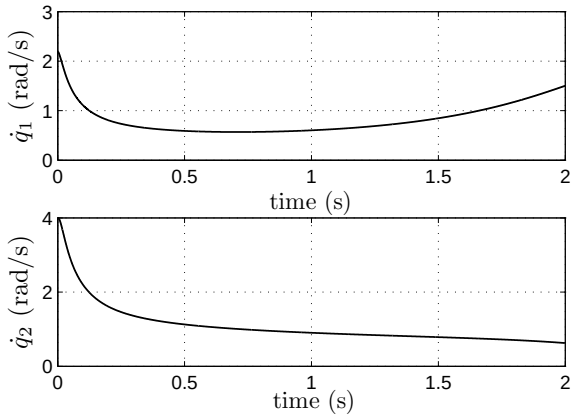


Fig. 6: Joint velocities' responses - simulation for $\hat{\mathbf{p}}_{o1}(0)$, $v_d = 0.25$ m/s

system. In the experiment, we actuate the second and fifth joints (red cylinders in Fig. 3), and start the experiment with the end-effector firmly gripping the handle of a cupboard door. The cupboard door is a 60 cm width IKEA kitchen cupboard, with multiple-link hinges, so that the centre of rotation moves slightly (<1 cm) as a function of door angle. The handle of the door has been extended an additional 5 cm to accomodate the width of the fingers on the parallel gripper. The DH parameters of the 7 DoF arm are the same as in simulation, see Table I. The two different scenarios based on different initial estimates of the radial direction (with errors of 50^o and 5^o, respectively) as well as different desired force/velocity values (v_d=0.05 m/s and v_d=0.1 m/s respectively) are given in Section V along the controller gains. The same gains as in simulation were used, and these have not been tuned specifically for the robot configuration or problem parameters, in order to show the generality of the approach. Figure 9 shows the robot performing the motion in the second case, with v_d=0.1 m/s.

The experimental results are shown in Figs. 10 and 11 for higher and lower uncertainty respectively; both force error

Fig. 9: The robot performing the door opening experiment. v_d = 0.1 m/s. The images are taken at $t = 0$ s, $t = 1.5$ s, $t = 3.6$ s, and $t = 5.7$ s, respectively.

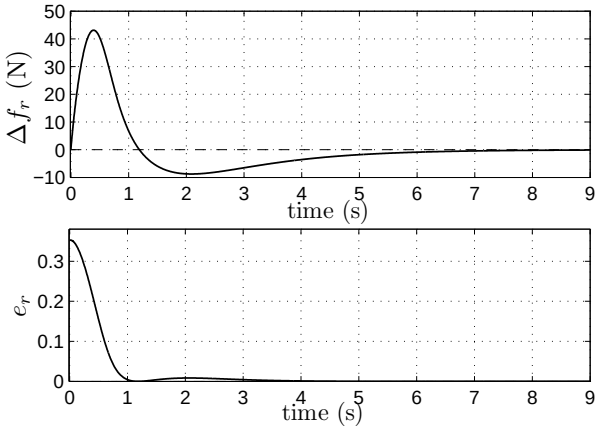


Fig. 7: Radial force error (upper plot) and estimation error (lower plot) responses - simulation with higher error in initial estimate $\hat{\mathbf{p}}_{o1}(0)$

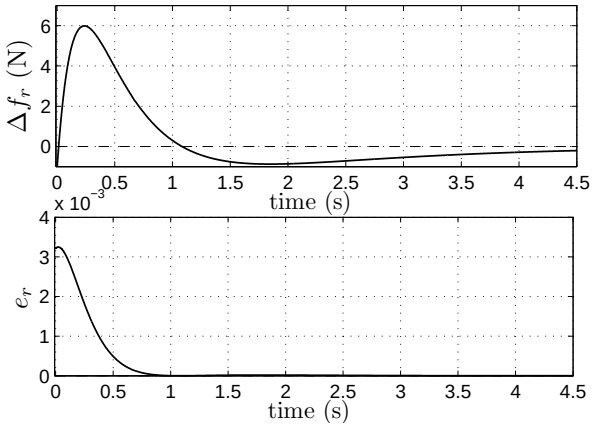


Fig. 8: Radial force error (upper plot) and estimation error (lower plot) responses - simulation with smaller error in initial estimate $\hat{\mathbf{p}}_{o2}(0)$

and estimation error converge to zero in approximately 2 s. In the real experiment, we see larger initial force errors and slower convergence than in simulation. This is to be expected, as the real experiment differs from the simulation in several aspects. The real experiment includes measurement noise and process noise, as well as communication delays. Also, since the manipulator was moved into the initial position using feed-forward position control, the initial state included larger force errors.

VII. Conclusions

This paper proposes a method for manipulation with uncertain kinematic constraints. It is inherently on-line and real-time, and convergence and stability is analytically provable. The method can be used with any velocity controllable manipulator with force measurements in the end-effector frame. In this paper, the method has been applied to the task of opening a door with unknown location of the hinges, while limiting the interaction forces.

The method consists of an adaptive controller which uses force and position/velocity measurements to deal with the door opening problem in the presence of incomplete knowledge of the door model. The controller uses an adaptive estimator of the door hinge's position to obtain adaptive estimates of the radial direction and to decompose the force and velocity control actions. The adaptive estimates of the radial direction are proven to converge to the actual radial vector, and the convergence of the radial force and the tangential velocity to the desired values has also been analytically proven. Simulation results along with an experiment on a real robot show that the estimates converge to the actual values even for large initial errors in the estimates, and the usefulness of the method has been demonstrated. Future work includes applying the proposed method to a wider range of domestic manipulation tasks with uncertainties in the kinematic constraints. Also, including humans in the loop and addressing human-robot collaborative manipulation will require extending the treatment to include dynamic uncertainties, and poses a challenging future problem.

Acknowledgment

This work has been supported by the Swedish Research Council (VR) and Swedish Foundation for Strategic Research (SSF).

References

[1] C. Cheah, C. Li, and J. Slotine, "Adaptive tracking control for robots with unknown kinematic and dynamic properties," *The International Journal of Robotics Research*, vol. 25, no. 3, pp. 283–296, 2006.

[2] C. C. Cheah, S. Kawamura, and S. Arimoto, "Stability of hybrid position and force control for robotic kinematics and dynamics uncertainties," *Automatica*, vol. 39, pp. 847–855, 2003.

[3] T. Yoshikawa, *Foundations of Robotics*. Cambridge, MA:MIT Press, 1990.

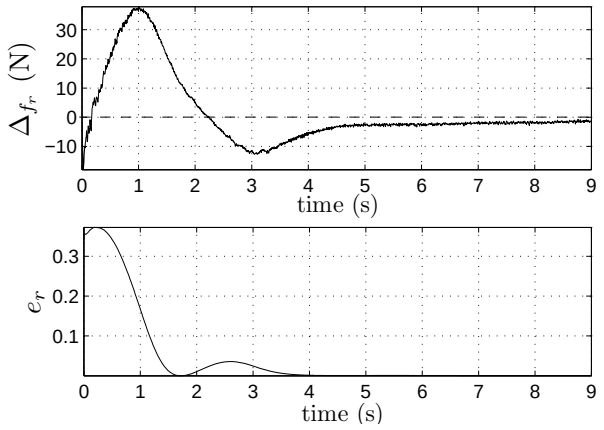


Fig. 10: Radial force (upper plot) and estimation error (lower plot) responses - robot experiment, higher error in initial estimate $\hat{\mathbf{p}}_{o1}(0)$

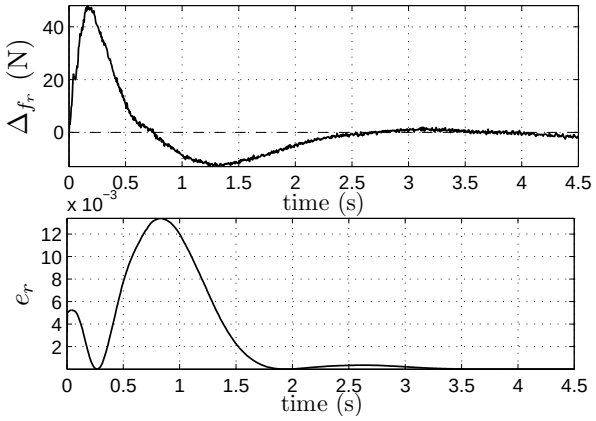


Fig. 11: Radial Force (Upper plot) and Estimation Error (Lower plot) Responses - robot experiment, smaller error in initial estimate $\hat{\mathbf{p}}_{o2}(0)$

[4] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators." *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, vol. 103, no. 2, pp. 126–133, 1981.

[5] Y. Karayiannidis and Z. Doulgeri, "Adaptive control of robot contact tasks with on-line learning of planar surfaces," *Automatica*, vol. 45, no. 10, pp. 2374–2382, 2009.

[6] K. Nagatani and S. Yuta, "An experiment on opening-door-behavior by an autonomous mobile robot with a manipulator," in *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems 95*, vol. 2, aug 1995, pp. 45 –50.

[7] G. Niemeyer and J.-J. Slotine, "A simple strategy for opening an unknown door," in *1997 IEEE International Conference on Robotics and Automation*, vol. 2, apr 1997, pp. 1448 –1453 vol.2.

[8] E. Lutscher, M. Lawitzky, G. Cheng, and S. Hirche, "A control strategy for operating unknown constrained mechanisms," in *IEEE International Conference on Robotics and Automation*, may 2010, pp. 819 –824.

[9] D. Ma, H. Wang, and W. Chen, "Unknown constrained mechanisms operation based on dynamic hybrid compliance control," in *IEEE International Conference on Robotics and Biomimetics*, dec. 2011, pp. 2366 – 2371.

[10] L. Peterson, D. Austin, and D. Kragic, "High-level control of a mobile manipulator for door opening," in *2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2000, pp. 2333 –2338.

[11] A. Jain and C. Kemp, "Pulling open novel doors and drawers with equilibrium point control," in *9th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2009*, dec. 2009, pp. 498 –505.

[12] ——, "Pulling open doors and drawers: Coordinating an omni-directional base and a compliant arm with equilibrium point control," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 1807–1814.

[13] M. Prats, S. Wieland, T. Asfour, A. del Pobil, and R. Dillmann, "Compliant interaction in household environments by the Armar-III humanoid robot," in *8th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2008*, dec. 2008, pp. 475 –480.

[14] H. Bruyninckx and J. De Schutter, "Specification of force-controlled actions in the "task frame formalism"-a synthesis," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 581–589, Aug 1996.

[15] W. Chung, C. Rhee, Y. Shim, H. Lee, and S. Park, "Door-opening control of a service robot using the multifingered robot hand," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 3975–3984, oct. 2009.

[16] J. Sturm, C. Stachniss, and W. Burgard, "A probabilistic framework for learning kinematic models of articulated objects," *Journal of Artificial Intelligence Research*, vol. 41, pp. 477–526, 2011.

[17] J. Sturm, A. Jain, C. Stachniss, C. Kemp, and W. Burgard, "Operating articulated objects based on experience," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, oct. 2010, pp. 2739 –2744.

[18] A. Petrovskaya and A. Y. Ng, "Probabilistic mobile manipulation in dynamic environments with application to opening doors," in *Proc. of the 20th International Joint Conference on Artifical Intelligence*, Hydrabad, India, January 2007, pp. 2178–2184.

[19] A. Schmid, N. Gorges, D. Goger, and H. Worn, "Opening a door with a humanoid robot using multi-sensory tactile feedback," in *IEEE International Conference on Robotics and Automation (ICRA)*, may 2008, pp. 285 –291.

[20] C. Kessens, J. Rice, D. Smith, S. Biggs, and R. Garcia, "Utilizing compliance to manipulate doors with unmodeled constraints," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, oct. 2010, pp. 483 –489.

[21] C. Ott, B. Bäuml, C. Borst, and G. Hirzinger, "Employing cartesian impedance control for the opening of a door: A case study in mobile manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on mobile manipulators: Basic techniques, new trends & applications*, 2005.

[22] D. Kim and G.-T. Kang, J.-H.and Park, "Door-opening behaviour by home service robot in a house," *International Journal of Robotics and Automation*, vol. 25, no. 4, pp. 271–284, 2010.

[23] H. Arisumi, J.-R. Chardonnet, and K. Yokoi, "Whole-body motion of a humanoid robot for passing through a door - opening a door by impulsive force -," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, oct. 2009, pp. 428 –434.

[24] P. A. Ioannou and J. Sun, *Robust Adaptive Control*. Upper Saddle River, NJ:Prentice Hall, 1996.

Model-free robot manipulation of doors and drawers by means of fixed-grasps

Yiannis Karayiannidis, Christian Smith, Francisco E. Viña, Petter Ögren, and Danica Kragic

***Abstract*— This paper addresses the problem of robot interaction with objects attached to the environment through joints such as doors, drawers and cupboards. We propose a methodology that requires no prior knowledge of the objects' kinematics, including the type of joint - either prismatic or revolute. The method consists of a velocity controller which relies on force/torque measurements and estimation of the motion direction, rotational axis and the distance from the center of rotation in the case of a hinged door. The method is suitable for any velocity controlled manipulator with a force/torque sensor at the end-effector. The force/torque control regulates the applied forces and torques within given constraints, while the velocity controller ensures that the end-effector of the robot moves with a task-related desired tangential velocity. The paper also provides a proof that the adaptive estimates converge to the actual values. The method is also evaluated in different scenarios, typically met in a household environment.**

I. Introduction

A robot operating in a domestic environment needs to interact with different types of doors, drawers and cupboards: objects having handles attached to them but also being attached to the other parts of the environment through joints. That is, the robot cannot perform a free manipulation on these but needs to take into account the external joint constraints. The variation in size, orientation and type of joints makes it intractable to provide a robot with predefined kinematic models of all doors it may encounter, and it is difficult - at times impossible - to observe and estimate the kinematic structure of a door or drawer before it is opened. This means that the performance of the task of opening different types of mechanisms can be significantly improved if the need to have prior knowledge of the mechanism is removed. In this paper, we propose a method for smooth, online opening of doors, drawers, or cupboards, without any need of prior knowledge of the mechanism.

Research on the door opening problem was formally initiated in the '90s with the experimental work on door opening [1] and a theoretically grounded work [2] proposing velocity-based estimation for the motion direction in door opening. Velocity-based estimation has inspired some of the recent works in opening domestic mechanisms such as doors and drawers [3], [4]. Velocity-based estimation is inherently online and allows the opening of a mechanism without explicit knowledge of its kinematic model or the kinematic parameters, but the proposed methods suffer from ill-defined normalization when the velocity is small and estimation lags.

The authors are with the Computer Vision and Active Perception Lab., Centre for Autonomous Systems, School of Computer Science and Communication, Royal Institute of Technology (KTH), SE-100 44 Stockholm, Sweden. e-mail: {yiankar|ccs|fevb|petter|dani}@kth.se

Position-based estimation techniques [5]–[8] that employ optimization algorithms working on end-effector position have also been used to estimate geometric characteristics of the mechanism rather than the motion direction. Since estimation does not guarantee identification in each control step, those methods have been coupled with controllers that provide the system with the proper compliance to absorb inaccuracies of the planned trajectories. On the other hand, probabilistic methods that are off-line and do not consider interaction force issues have been used for more advanced estimation tasks: in [9], the kinematic characteristics of complicated mechanisms with more DoFs are learned, while in [10] state estimation is studied given a detailed model of the door. Another part of the literature on the door opening problem exploits advanced hardware capabilities in order to open a door. In [11], [12], robot compliant behavior has been used to accomplish the manipulation task without estimating the direction of motion or the kinematics of the mechanism in hand. In [13], slowly pulling and pushing in a prior phase is used to estimate the kinematics of the mechanism by sensing through tactile sensors the forces exerted on the fingertips, while in [14], the objective is to exert an impulsive force on a swinging door.

In Table I, we summarize some defining characteristics of the door opening methods found in the literature and compare to the present paper. In the table, the term *force control* designates work that explicitly controls or limits the interaction forces, *online, real-time* implies that the method can be used to open a door directly — at human-like velocities — without any prior learning step, *moderate hardware requirements* means that the method can be used on a simple manipulator with velocity control and a force/torque sensor, and *revolute doors* and *sliding doors* describe what types of door kinematics that can be handled by the method. *Estimate of constraints* indicate methods that produce an estimate of the current kinematic constraints of a mechanism, while *estimate of geometry* indicate methods that produce an explicit estimate of the geometry of the door mechanics themselves. *Unknown model* indicates methods that will work properly even if the model (type of mechanism, i.e. revolute or prismatic joint) is not known á priori, and *unknown parameters* indicate methods that will work if the parameters of the mechanism (i.e. hinge position or motion axis of prismatic joint) are not known á priori. Finally, *proven parameter identification* indictates whether proofs are provided for the convergence of estimations.

In previous work, we have proposed a door opening control algorithm that produces estimates of the center of

TABLE I : Comparison of related works and this paper.

Publications	[1]	[2]	[3]	[4]	[5]	[6], [7]	[8]	[13]	[9]	[10]	[11]	[12]	[14]	[15], [16]	our approach
Force control	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓	✓
Online, real-time	✓	✓	✓	✓	✓	✓	✓				✓	✓	✓	✓	✓
Moderate H/W Spec.	✓	✓	✓	✓	✓	✓	✓	[1]	✓	✓	[2]	[3]	[4]	✓	✓
Revolute Doors	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Sliding Doors		✓	✓	✓		✓	✓		✓	✓	✓				✓
Estimate of Constraints		✓	✓	✓	✓	✓		✓	✓					✓	✓
Estimate of Geometry					✓	✓		✓	✓						✓
Unknown Model		✓	✓	✓		✓			✓		✓		✓		✓
Unknown Parameters		✓	✓	✓	✓	✓		✓	✓				✓	✓	✓
Proven Parameter Identification														✓	✓

[1] Multifingered hand with tactile sensors [2] Compliant joints (torque feedback at the joint level) – DLR lightweight robot II
[3] Joint compliance by using clutches to engage/disengage motors [4] Use of the humanoid robot HRP-2

rotation for a revolute door, exploits the estimates in the proposed velocity reference and which is proved to identify the constraint direction as well as achieve velocity/force tracking for smooth door opening [15], [16]. The control scheme assumes a known kinematic model for the door — revolute — but the center of rotation is considered uncertain. Furthermore, the estimator uses a projection operator to guarantee well-defined updated estimates; the use of a projection set constrains the range of uncertainties that can be dealt with. In contrast to the previous work, we now propose a control scheme that can deal with both revolute-joint doors and sliding doors/drawers by constructively utilizing the fixed-grasp assumption. Furthermore, the design of the estimator does not require a projector operator; the estimator produces inherently well-defined estimates that converge to the actual values. In the following list we summarize our contribution as compared to the existing literature:
— Our method can be applied to open both rotational and sliding doors, without requiring ill-defined normalization.
— Our method is not based on unusual hardware capabilities and can be implemented in any velocity controlled manipulator with a forque/torque sensor at the wrist.
— Our method can be proved theoretically to achieve identification of the motion direction simultaneously with force/velocity convergence by explicitly considering adaptive estimates in the controller design.

The remainder of this paper is organised as follows: Section II provides description of the notation, system kinematics, a preliminary control example and problem formulation. The proposed solution and the corresponding stability analysis are given in Section III followed by the evaluation in simulation in Section IV. In Section V the final outcome of this work is briefly discussed.

II. System and Problem Description

Generally, doors and drawers can be opened by grasping the handle and moving this along its intended trajectory of motion, which would be a along a circular path for hinged mechanisms, or along a linear path for sliding doors and drawers. In this section we formally define the problem of door/drawer opening under uncertainty, where the position of hinges, or direction of possible sliding motion is not known á priori.

A. Notation and Preliminaries

We introduce the following notation:

- Bold small letters denote vectors while bold capital letters denote matrices. Underline $\underline{\cdot}$, hat $\widehat{\cdot}$ and tilde $\tilde{\cdot}$ are used for denoting vectors of unit magnitude, estimates and errors between control variables and their corresponding desired values/vectors respectively. $\cdot^\top$ denotes the transpose of a vector or a matrix.
- The generalized position of a frame $\{i\}$ with respect to a frame $\{j\}$ is described by a position vector ${}^j\mathbf{p}_i \in \mathbb{R}^m$ and a rotation matrix ${}^j\mathbf{R}_i \in SO(m)$ where $m = 2$ or 3 for the planar and spatial case respectively. In case $\{j\} \equiv \{B\}$ where $\{B\}$ is the robot world inertial frame (located usually at the robots base) the left superscript is omitted. Each column of ${}^j\mathbf{R}_i$ is denoted by ${}^j\underline{\mathbf{x}}_i \equiv \mathbf{R}_j^\top \underline{\mathbf{x}}_i$, ${}^j\underline{\mathbf{y}}_i \equiv \mathbf{R}_j^\top \underline{\mathbf{y}}_i$, ${}^j\underline{\mathbf{z}}_i \equiv \mathbf{R}_j^\top \underline{\mathbf{z}}_i$ where $\underline{\mathbf{x}}_i$, $\underline{\mathbf{y}}_i$, $\underline{\mathbf{z}}_i$ denote the columns of the rotation matrix $\mathbf{R}_i$ that describes the orientation of the frame $\{i\}$ with respect to the robot world inertial frame.
- The projection matrix on a unit three dimensional vector $\mathbf{a}$ is denoted by $\mathbf{P}(\mathbf{a})$, and is defined as follows:
$$\mathbf{P}(\mathbf{a}) = \mathbf{a}\mathbf{a}^\top$$
while the projection matrix on $\mathbf{a}$'s orthogonal complement space is denoted by $\bar{\mathbf{P}}(\mathbf{a})$ and can be defined as follows:
$$\bar{\mathbf{P}}(\mathbf{a}) = \mathbf{I}_3 - \mathbf{P}(\mathbf{a})$$
- We denote with $\mathcal{I}(\mathbf{b})$ the element-wise integral of some vector function of time $\mathbf{b}(t) \in \mathbb{R}^n$ over the time variable t, i.e:
$$\mathcal{I}(\mathbf{b}) = \int_0^t \mathbf{b}(\tau) d\tau$$
- We denote with $\mathbf{S}(\mathbf{b})$ the skew-symmetric matrix produced by $\mathbf{b} \triangleq [b_x\ b_y\ b_z]^\top$ as follows:
$$\mathbf{S}(\mathbf{b}) = \begin{bmatrix} 0 & -b_z & b_y \\ b_z & 0 & -b_x \\ -b_y & b_x & 0 \end{bmatrix}$$
in order to perform a cross product operation with any three-dimensional vector $\mathbf{a} \in \mathbb{R}^3$ i.e. $\mathbf{b} \times \mathbf{a} = \mathbf{S}(\mathbf{b})\mathbf{a}$.

Note that $\mathbf{S}(\mathbf{b})\mathbf{a} = -\mathbf{S}(\mathbf{a})\mathbf{b}$. Note also that $\forall \mathbf{R} \in SO(3)$ the following similarity transformation holds: $\mathbf{S}(\mathbf{R}\mathbf{b}) = \mathbf{R}\mathbf{S}(\mathbf{b})\mathbf{R}^\top$.

B. Kinematic model of robot door/drawer opening

We consider a setting of a robot manipulator in which its end-effector has achieved a fixed grasp of the handle of a mechanism kinematically modeled as revolute or prismatic joint e.g. a door or a drawer in a domestic environment. The term fixed grasp describes that there is neither relative translational motion between the handle and the end-effector nor relative rotation of the end-effector around the handle.

Let $\{e\}$ and $\{h\}$ be the end-effector and the handle frame respectively. The fixed grasp assumption implies the invariance of the relative position and orientation of the aforementioned frames, formally expressed as follows:

$$ {}^e\dot{\mathbf{p}}_h = 0, \quad {}^e\dot{\mathbf{R}}_h = 0 \tag{1}$$

The two frames are attached on a kinematically known position e.g. a known point of the end-effector denoted by $\mathbf{p}_e$. However, the end-effector and handle frames are described by different rotation matrices since they are strictly connected to the robot kinematics and the door/drawer kinematics respectively. In case of a rotating door (revolute joint) we also consider a frame $\{o\}$ attached at the center of the circular trajectory of the end-effector while opening the rotating door. The axis $\underline{\mathbf{z}}_o$ corresponds to the axis of the rotation while $\underline{\mathbf{x}}_o$, $\underline{\mathbf{y}}_o$ can be arbitrarily chosen (Fig. 1i).

In the following we state a convention in order to define the frame $\{h\}$ in both cases of revolute joints (hinged doors) and prismatic joints (sliding doors, drawers):

Revolute joints

- Axis $\underline{\mathbf{z}}_h$ is equivalent to $\underline{\mathbf{z}}_o$, i.e. $\underline{\mathbf{z}}_o \equiv \underline{\mathbf{z}}_h$
- Axis $\underline{\mathbf{y}}_h$ is the unit vector which is perpendicular to both $\underline{\mathbf{z}}_h$ and $\underline{\mathbf{z}}_o$ with direction towards the hinge.
- Axis $\underline{\mathbf{x}}_h$ can be regarded as the allowed motion axis; it can be formed as follows: $\underline{\mathbf{x}}_h = \underline{\mathbf{y}}_h \times \underline{\mathbf{z}}_h = \mathbf{S}(\underline{\mathbf{y}}_h)\underline{\mathbf{z}}_h$.

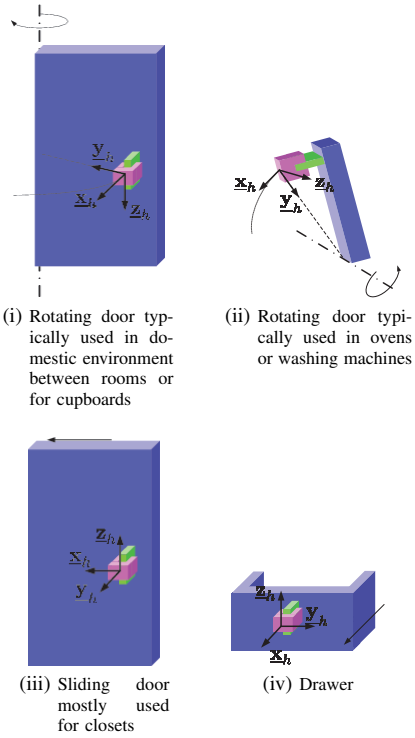
Prismatic joints

Vector $\underline{\mathbf{x}}_h$ denotes the allowed motion axis. Axes $\underline{\mathbf{z}}_h$ and $\underline{\mathbf{y}}_h$ can be arbitrarily chosen in order to span the two-dimensional surface to which $\underline{\mathbf{x}}_h$ is perpendicular.

Examples of Fig. 1 illustrate the definition of the $\{h\}$ axes. Note that the $\{h\}$ axes definition is based on door/drawer kinematics that are uncertain in an unknown environment (e.g. domestic environment) in which the robot can identify and grasp handles of doors and drawers but it cannot perceive the kinematics of their mechanism.

In case of a door kinematically modeled as a revolute joint, we can define the radial vector –which is parallel to $\underline{\mathbf{y}}_h$– as the relative position of the frames $\{o\}$ and $\{e\}$:

$$ \mathbf{r} \triangleq \mathbf{p}_o - \mathbf{p}_e \tag{2}$$



(i) Rotating door typically used in domestic environment between rooms or for cupboards

(ii) Rotating door typically used in ovens or washing machines

(iii) Sliding door mostly used for closets

(iv) Drawer

Fig. 1 : Illustrative examples of different types of rotating/sliding doors and drawers that can be modeled as revolute and prismatic joints

By expressing $\mathbf{r}$ with respect to the end effector frame and differentiating the resultant equation we get:

$$ \dot{\mathbf{R}}_e {}^e\mathbf{r} + \mathbf{R}_e {}^e\dot{\mathbf{r}} = \dot{\mathbf{p}}_o - \dot{\mathbf{p}}_e \tag{3}$$

Notice that ${}^e\dot{\mathbf{r}} = {}^e\dot{\mathbf{R}}_h {}^h\mathbf{r} + {}^e\mathbf{R}_h {}^h\dot{\mathbf{r}} = 0$ since ${}^e\dot{\mathbf{R}}_h = 0$ and ${}^h\dot{\mathbf{r}} = 0$ are implied by the fixed grasp assumption. By substituting ${}^e\dot{\mathbf{r}} = \dot{\mathbf{p}}_o = 0$ as well as $\dot{\mathbf{R}}_e\mathbf{R}_e^\top = \mathbf{S}(\boldsymbol{\omega}_e)$ with $\boldsymbol{\omega}_e$ being the rotational velocity of the end-effector, we get:

$$ \dot{\mathbf{p}}_e = \mathbf{S}(\mathbf{r})\boldsymbol{\omega}_e \tag{4}$$

which describes the first-order differential kinematics of the door opening in case of a revolute hinge. By multiplying both sides of (4) with $\mathbf{R}_e^\top$ and using the similarity transformation for the skew-symmetric matrix $\mathbf{S}(\mathbf{r})$ we can express the constraint in the end-effect frame as follows:

$$ \mathbf{v} = \mathbf{S}({}^e\mathbf{r})\boldsymbol{\omega} \tag{5}$$

with $\mathbf{v} = {}^e\dot{\mathbf{p}}_e = \mathbf{R}_e^\top\dot{\mathbf{p}}_e$ and $\boldsymbol{\omega} = {}^e\boldsymbol{\omega}_e = \mathbf{R}_e^\top\boldsymbol{\omega}_e$ are the translational and rotational end-effector velocities expressed in the local frame of the end-effector. By denoting with ℓ the distance between the end-effector frame and the center of rotation, the radial vector ${}^e\mathbf{r}$ can be written as follows:

$$ {}^e\mathbf{r} = \ell\mathbf{R}_e^\top\underline{\mathbf{y}}_h = \ell {}^e\underline{\mathbf{y}}_h \tag{6}$$

By taking the inner product of (5) with ${}^e\underline{\mathbf{x}}_h$ and using (6) we get:

$$ {}^e\underline{\mathbf{x}}_h^\top\mathbf{v} = \ell {}^e\underline{\mathbf{z}}_h^\top\boldsymbol{\omega} \tag{7}$$

The remaining constraints, that stem from the fixed grasp assumption, i.e. $\mathbf{v} - {}^e\dot{\mathbf{p}}_h = 0$ and $\boldsymbol{\omega}_e - \boldsymbol{\omega}_h = 0$, can be imposed to the end-effector translational and rotational velocities as follows:

$$\bar{\mathbf{P}}({}^e\underline{\mathbf{x}}_h)\mathbf{v} = 0 \tag{8}$$

$$\bar{\mathbf{P}}({}^e\underline{\mathbf{z}}_h)\boldsymbol{\omega} = 0 \tag{9}$$

Notice that one way to express the four constraints imposed from the equations (8), (9) is the following:

$${}^e\underline{\mathbf{y}}_h^\top\mathbf{v} = 0 \qquad {}^e\underline{\mathbf{z}}_h^\top\mathbf{v} = 0 \tag{10}$$

$${}^e\underline{\mathbf{x}}_h^\top\boldsymbol{\omega} = 0 \qquad {}^e\underline{\mathbf{y}}_h^\top\boldsymbol{\omega} = 0 \tag{11}$$

Constraints (10) and (11) have been derived from (8) and (9) respectively.

In case of a sliding door or a drawer the fixed grasp assumption implies that $\boldsymbol{\omega}_e = \boldsymbol{\omega}_h = 0$. Hence, the constraint (7) related to the rotational motion of the end-effector can be replaced by:

$${}^e\underline{\mathbf{z}}_h^\top\boldsymbol{\omega} = 0 \tag{12}$$

The remaining constraints [(8),(9) or (10),(11)] remain valid for the prismatic case as well.

C. Robot kinematic model

In case of velocity controlled manipulators, the robot joint velocity is controlled directly by the reference velocity $\mathbf{u}_{\text{ref}}$. In particular, the reference generalized velocity $\mathbf{u}_{\text{ref}} \triangleq [\mathbf{v}_{\text{ref}}^\top \ \boldsymbol{\omega}_{\text{ref}}^\top]^\top \in \mathbb{R}^6$ ($\mathbf{v}_{\text{ref}} \in \mathbb{R}^3$ and $\boldsymbol{\omega}_{\text{ref}} \in \mathbb{R}^3$ denote the translational and rotational part respectively) expressed at the end-effector frame can be considered as a kinematic controller which is mapped to the joint space in order to be applied at the joint velocity level as follows:

$$\dot{\mathbf{q}} = \mathbf{J}^+(\mathbf{q})\boldsymbol{\Gamma}(\mathbf{q})\mathbf{u}_{\text{ref}} \tag{13}$$

with:

- $\mathbf{q},\ \dot{\mathbf{q}} \in \mathbb{R}^n$ being the joint positions and velocities respectively.
- $\mathbf{J}(\mathbf{q})^+ = \mathbf{J}(\mathbf{q})^\top\left[\mathbf{J}(\mathbf{q})\mathbf{J}(\mathbf{q})^\top\right]^{-1}$ being the pseudo-inverse of the manipulator Jacobian $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{6\times n}$ which relates the joint velocities $\dot{\mathbf{q}}$ to the end-effector velocities $[\dot{\mathbf{p}}_e^\top \ \boldsymbol{\omega}_e^\top]^\top$
- $\boldsymbol{\Gamma}(\mathbf{q}) = \text{diag}_{\text{block}}[\mathbf{R}_e, \mathbf{R}_e]$ being a transformation for mapping the velocity from end-effector frame to the global inertial frame.

If we consider the typical Euler-Lagrange robot dynamic model, the velocity error at the joint level drive the torque (current) controller $\mathbf{u}_{\boldsymbol{\tau}}(t)$. If we assume a high frequency current control loop with external forces' compensators and weak inertial dynamics, then the kinematic model is valid.

D. Stability of a simple non-autonomous system using a logarithmic barrier function

In this section we will state and prove a lemma to be used in the proof of the main result of the paper.

Lemma 1: Consider the state domain $D \triangleq \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ and the non-autonomous system (dependent on time variable and state θ) defined in D and described by the following differential equation:

$$\dot{\theta} = -\gamma c(t)\tan\theta \tag{14}$$

with γ and $c(t)$ being a strictly positive constant and a non-negative function of t respectively.

If $c(t)$ satisfies the persistent excitation condition i.e. $\int_t^{t+T_0} c(\tau)d\tau \geq \alpha_0 T_0$ for some α_0 and T_0, then $\theta(t)$ converges to zero exponentially, and

$$\theta(t) = \arcsin\left[e^{-\gamma\int_0^t c(\tau)d\tau}\theta(0)\right], \tag{15}$$

Note that a non-negative sinusoidal or step-function satisfies the aforementioned condition.

Proof: Consider the function $U(\theta) : D \to \mathbb{R}^+$, given by:

$$U(\theta) = -\ln(\frac{1}{\gamma}\cos\theta) \tag{16}$$

Differentiating $U(\theta)$ with respect to time and substituting (14) we get:

$$\dot{U}(\theta, t) = -c(t)\tan^2\theta \tag{17}$$

Since $c(t) \geq 0$, $\dot{U}(\theta, t) \leq 0$ which in turn implies $U(\theta) \leq U(\theta(0))$ and $\theta(t) \in D$, $\forall t$ for $\theta(0) \in D$, we can express the system using the variable $\sigma = \sin\theta$ as follows: $\dot{\sigma} = -\gamma c(t)\sigma$ and subsequently calculate the analytic solution which is given by: $\theta(t) = \arcsin\left[e^{-\gamma\int_0^t c(\tau)d\tau}\theta(0)\right]$ which implies the convergence stated above. ■

E. Control Objective

Our target is to control the motion of the robot to manipulate and open an external mechanism, such as a door or drawer, irrespective of its kinematic structure. In applications for dynamic unstructured — e.g. domestic — environments, it is difficult to design á priori the desired velocity satisfying the constraints imposed by the mechanism. This is due to the difficulties of identifying the kinematic characteristics of the mechanism. The execution of a trajectory incompatible with system constraints gives rise to high interaction forces which may be harmful to both the manipulated mechanism and the robot, and does not lead to a successful task accomplishment.

The task can be naturally described in the handle frame, but the desired variables should be defined at the end-effector frame to be executable by the robot. Let $\mathbf{f}_d$, $\boldsymbol{\tau}_d$ and $\mathbf{v}_d(t)$ be the desired force, torque and velocity expressed at the end-effector frame respectively. Let $v_d(t)$ be the desired velocity along the motion axis of frame $\{h\}$. Then the desired velocity at the end-effector frame is defined along ${}^e\underline{\mathbf{x}}_h$, i.e. $\mathbf{v}_d = {}^e\underline{\mathbf{x}}_h v_d(t)$, and the force control objective can be achieved by projecting the desired force on the orthogonal complement space of ${}^e\underline{\mathbf{x}}_h$ (constrained directions) i.e. $\bar{\mathbf{P}}({}^e\underline{\mathbf{x}}_h)\mathbf{f}_d$; a small valued or zero vector $\mathbf{f}_d$ corresponds to small forces along the constraint directions. On the other hand, the desired rotational velocity can be defined using $v_d(t)$ along the axis $\mathbf{d} \triangleq d\,{}^e\underline{\mathbf{z}}_h$, i.e. $\boldsymbol{\omega}_d(t) = \mathbf{d}v_d(t)$, with:

$$d \triangleq \begin{cases} 1/\ell, & \text{for rotational mechanisms} \\ 0, & \text{for sliding mechanisms} \end{cases} \tag{18}$$

Note that prismatic kinematics can be approximated by rotational kinematics using large values of ℓ .

If we denote the total interaction force and torque expressed at the end-effector frame with $\mathbf{f} \in \mathbb{R}^3$ and $\boldsymbol{\tau} \in \mathbb{R}^3$ respectively the control objective can be formulated as:

Problem 1 (Door/Drawer Opening Problem): Design a velocity control $\mathbf{u}_{\text{ref}}$ such that $\bar{\mathbf{P}}({}^e\underline{\mathbf{x}}_h)\mathbf{f} \rightarrow \bar{\mathbf{P}}({}^e\underline{\mathbf{x}}_h)\mathbf{f}_d$, $\boldsymbol{\tau} \rightarrow \boldsymbol{\tau}_d$, $\mathbf{v} \rightarrow {}^e\underline{\mathbf{x}}_h v_d(t)$, $\boldsymbol{\omega} \rightarrow d^e\underline{\mathbf{z}}_h v_d(t)$, without knowing accurately the motion axis ${}^e\underline{\mathbf{x}}_h$, the corresponding constraint directions $\bar{\mathbf{P}}({}^e\underline{\mathbf{x}}_h)$, or the axis of rotation ${}^e\underline{\mathbf{z}}_h$ and the variable d. From a high level perspective, we consider that the opening task is accomplished when the observed end-effector trajectory — which coincides with the handle trajectory — has progressed far enough to enable the robot to perform a subsequent task, like picking up an object or passing through a door. Hence, some perception system observing the progress of the opening of the mechanism is additionally required to provide the robot with the command to halt the opening procedure.

III. Control Design

In this section, we will propose a solution to Problem 1 from Section II-E. First, Properties 1 and 2 show that estimator (25) correctly identifies the direction of motion, then Theorems 1 and 2 show that a complete solution to the problem has been proposed in (19)-(25) and (32).

A. Translational velocity reference with torque feedback

Let ${}^e\widehat{\underline{\mathbf{x}}}_h(t)$ denote the online estimate of motion direction ${}^e\underline{\mathbf{x}}_h$. Dropping the argument t from ${}^e\widehat{\underline{\mathbf{x}}}_h(t)$and $v_d(t)$ for notation convenience we introduce a reference velocity vector $\mathbf{v}_{\text{ref}}$ for controlling the end-effector translational velocity:

$$\mathbf{v}_{\text{ref}} = {}^e\widehat{\underline{\mathbf{x}}}_h v_d - \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f \tag{19}$$

where $\mathbf{v}_f$ is a PI force feedback input defined as follows:

$$\mathbf{v}_f = \alpha_f \tilde{\mathbf{f}} \ + \beta_f \mathcal{I}\left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}}\right] \tag{20}$$

with $\tilde{\mathbf{f}} = \mathbf{f} - \mathbf{f}_d$ and α_f, β_f being positive control constants.

Let $\theta(t)$ denotes the angle formed between the actual vector ${}^e\underline{\mathbf{x}}_h$ and its online estimate ${}^e\widehat{\underline{\mathbf{x}}}_h$ which is time-varying. Given that the estimate ${}^e\widehat{\underline{\mathbf{x}}}_h$ is a unit vector, $\cos\theta(t)$ can be defined as follows:

$$\cos\theta(t) = {}^e\underline{\mathbf{x}}_h^\top {}^e\widehat{\underline{\mathbf{x}}}_h \tag{21}$$

In general, an online estimate of the vector ${}^e\widehat{\underline{\mathbf{x}}}_h$ provided by an adaptive estimator is not unit but in the following we are going to design an update law that produces estimates of unit magnitude. In the following, we drop out the argument of t from $\theta(t)$ for notation convenience.

The velocity error $\tilde{\mathbf{v}} \triangleq \mathbf{v} - \mathbf{v}_{\text{ref}}$ can be decomposed along ${}^e\widehat{\underline{\mathbf{x}}}_h$ and the corresponding orthogonal complement space as follows:

$$\tilde{\mathbf{v}} = \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)(\mathbf{v} + \mathbf{v}_f) + {}^e\widehat{\underline{\mathbf{x}}}_h \left(v\cos\theta(t) - v_d\right) \tag{22}$$

where v denotes the magnitude of the velocity. In case of velocity controlled manipulators, it is assumed that $\tilde{\mathbf{v}} = 0$ (13) which implies the following closed-loop system equations:

$$\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f = -\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h){}^e\underline{\mathbf{x}}_h v \tag{23}$$

$$v = \tfrac{1}{\cos\theta(t)} v_d \tag{24}$$

We design the update law for ${}^e\widehat{\underline{\mathbf{x}}}_h$ as follows:

$${}^e\dot{\widehat{\underline{\mathbf{x}}}}_h = -\gamma v_d \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f \tag{25}$$

The use of the update law (25) is instrumental for the stability analysis and the convergence proof. Furthermore, update law (25) has two basic properties:

Property 1: The update law (25) ensures that the norm of ${}^e\widehat{\underline{\mathbf{x}}}_h(t)$ is invariant, i.e. starting with $\|{}^e\widehat{\underline{\mathbf{x}}}_h(0)\| = 1$, $\|{}^e\widehat{\underline{\mathbf{x}}}_h(t)\| = 1, \ \forall t$.

Proof: By projecting (25) along ${}^e\widehat{\underline{\mathbf{x}}}_h$ yields $\frac{d}{dt}\left(\|{}^e\widehat{\underline{\mathbf{x}}}_h(t)\|^2\right) = -\gamma v_d \left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h){}^e\widehat{\underline{\mathbf{x}}}_h\right]^\top \mathbf{v}_f = 0$ (since $\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h){}^e\widehat{\underline{\mathbf{x}}}_h = 0$). ■

Property 2: The update law (25) yields the scalar differential equation (14) with respect to angle θ defined in (21) with $c(t) := v_d^2 \geq 0$, i.e., the estimate converges to the true vector.

Proof: The second property is proven by projecting both sides of (25) along ${}^e\underline{\mathbf{x}}_h$: Substituting (21) in the left side of the projected (25) yields:

$${}^e\underline{\mathbf{x}}_h^\top {}^e\dot{\widehat{\underline{\mathbf{x}}}}_h = \tfrac{d}{dt}(\cos\theta) = -\sin\theta\dot{\theta} \tag{26}$$

Substituting (23), (24) and ${}^e\underline{\mathbf{x}}_h^\top \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h){}^e\underline{\mathbf{x}}_h = \sin^2\theta$ in the right side of the projected (25) yields:

$$-\gamma v_d {}^e\underline{\mathbf{x}}_h^\top \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f = \tfrac{\gamma v_d^2}{\cos\theta} {}^e\underline{\mathbf{x}}_h^\top \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h){}^e\underline{\mathbf{x}}_h = \gamma v_d^2 \sin\theta\tan\theta \tag{27}$$

By combining (26) and (27) we get:

$$\sin\theta(\dot{\theta} + \gamma v_d^2 \tan\theta) = 0 \tag{28}$$

Eq. (28) has a trivial solution $\theta(t) = 0$ and the solution and the solution given by Lemma 1 in equation (14) with $c(t) = v_d^2$ which includes the trivial one. ■

Using the aforementioned properties of the update law (25), we can prove the following Theorem:

Theorem 1: Consider a velocity controlled manipulator, with first order differential kinematics described by (13), which has achieved a fixed grasp with the handle of a sliding/rotating door or a drawer.

If the robot is driven by a velocity control input $\mathbf{v}_{\text{ref}}$ (19) that uses a PI force feedback input $\mathbf{v}_f$ (20) as well as the update law (25) to estimate the local motion axis ${}^e\underline{\mathbf{x}}_h$, then a subset of Problem 1 will be solved, i.e., smooth opening of the moving mechanism will be achieved. Analytically, the following convergence results are guaranteed: ${}^e\widehat{\underline{\mathbf{x}}}_h \rightarrow {}^e\underline{\mathbf{x}}_h$, $\mathbf{v} \rightarrow {}^e\underline{\mathbf{x}}_h v_d$, $\bar{\mathbf{P}}({}^e\underline{\mathbf{x}}_h)\tilde{\mathbf{f}} \rightarrow 0$, given that v_d is appropriately chosen.

Proof: Consider the following Lyapunov-like function:

$$V = \alpha_f \beta_f \|\mathcal{I}\left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}}\right]\|^2 + U(\theta) \tag{29}$$

with $U(\theta)$ being defined in (16). By differentiating (29), adding and subtracting $\alpha_f^2 \left\| \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}} \right\|^2$, $\beta_f^2 \left\| \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathcal{I}\left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}}\right] \right\|^2$ and subsequently substituting (23), (24) we get:

$$\dot{V} = -\alpha_f^2 \left\| \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}} \right\|^2 - \beta_f^2 \left\| \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathcal{I}\left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}}\right] \right\|^2 - \frac{v_d}{\cos\theta}\mathbf{v}_f^\top \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)^e\underline{\mathbf{x}}_h - \frac{1}{\gamma\cos\theta}{}^e\dot{\widehat{\underline{\mathbf{x}}}}_h^\top {}^e\underline{\mathbf{x}}_h \quad (30)$$

In order to cancel out the terms of the second line in (30) we substitute the update law (25).

Consequently, the derivative of function V (29) along the system trajectories (23), (24) and (25) is given by:

$$\dot{V} = -\alpha_f^2 \left\| \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}} \right\|^2 - \beta_f^2 \left\| \bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathcal{I}\left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}}\right] \right\|^2$$

Hence, $V(t) \leq V(0)$, $\forall\, t$ which implies that $\mathcal{I}(\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}})$ is bounded and that $\theta(t) \in D$ provided that $\theta(0) \in D$ (For details see Section II-D). Consequently, (23), (24) implies that $\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f$ and v are bounded. Furthermore, the boundedness $\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f$ implies that the update law rate ${}^e\dot{\widehat{\underline{\mathbf{x}}}}_h$ is bounded. Differentiating (23), (24) and using the boundedness of $\mathcal{I}[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}}]$, $\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f$ and ${}^e\dot{\widehat{\underline{\mathbf{x}}}}_h$, it can be easily shown that $\frac{d}{dt}\left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathbf{v}_f\right]$ is bounded. Hence, the second derivative of V is bounded allowing the use of Barbalat's Lemma in order to prove that $\dot{V} \to 0$ and consequently $\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\mathcal{I}\left[\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}}\right]$, $\bar{\mathbf{P}}({}^e\widehat{\underline{\mathbf{x}}}_h)\tilde{\mathbf{f}} \to 0$. Note that the aforementioned convergence results are referred to the estimated motion space defined by ${}^e\widehat{\underline{\mathbf{x}}}_h$. By using the Property 2 of the update law, (25) yields (14) and subsequently (15) that implies the exponential convergence of θ to zero or ${}^e\widehat{\underline{\mathbf{x}}}_h \to {}^e\underline{\mathbf{x}}_h$ for v_d satisfying the persistent excitation condition. ■

B. Rotational velocity with torque feedback

Since the translational velocity is strictly connected to the rotational velocity, the reference rotational velocity will be defined using the desired translational velocity v_d as follows:

$$\boldsymbol{\omega}_{\text{ref}} = \widehat{\mathbf{d}} v_d - \boldsymbol{\omega}_\tau \quad (31)$$

where $\widehat{\mathbf{d}}$ is the online estimate of $\mathbf{d}$ and it is appropriately designed as follows:

$$\dot{\widehat{\mathbf{d}}} = -\gamma_d v_d \boldsymbol{\omega}_\tau \quad (32)$$

and $\boldsymbol{\omega}_\tau$ is a PI torque feedback input defined as follows:

$$\boldsymbol{\omega}_\tau = \alpha_\tau \tilde{\boldsymbol{\tau}} \; + \beta_\tau \mathcal{I}(\tilde{\boldsymbol{\tau}}) \quad (33)$$

The design of the update law (32) is instrumental for the proof of the following theorem:

Theorem 2: Consider a velocity controlled manipulator, with first order differential kinematics described by (13), which has achieved a fixed grasp with the handle of a sliding/rotating door or a drawer. If the robot is driven by a velocity control input that consists of both $\mathbf{v}_{\text{ref}}$ (19) and $\boldsymbol{\omega}_{\text{ref}}$ (31) that uses a PI torque feedback input $\boldsymbol{\omega}_\tau$ (33) as well as the update law (32) to estimate the vector $\mathbf{d}$, then Problem 1 will be entirely solved, i.e., the following convergence results –additionally to those of Theorem 1– are guaranteed: $\tilde{\boldsymbol{\tau}} \to 0$, $\mathcal{I}(\tilde{\boldsymbol{\tau}}) \to 0$, $\boldsymbol{\omega} \to \mathbf{d} v_d$, for an appropriately chosen v_d.

Proof: First, we will reform $\boldsymbol{\omega}_{\text{ref}}$ by adding/subtracting the term $\mathbf{d}(v - v_d)$ and using (24) as follows:

$$\boldsymbol{\omega}_{\text{ref}} = \mathbf{d} v + \tilde{\mathbf{d}} v_d - \boldsymbol{\omega}_\tau + \mathbf{d}\left(\tfrac{\cos\theta - 1}{\cos\theta}\right) v_d \quad (34)$$

with $\tilde{\mathbf{d}} = \widehat{\mathbf{d}} - \mathbf{d}$. For design purposes we consider the following positive definite function:

$$W = \alpha_\tau \beta_\tau \|\mathcal{I}(\tilde{\boldsymbol{\tau}})\|^2 + \frac{1}{2\gamma_d}\|\tilde{\mathbf{d}}\|^2 + \xi U(\theta) \quad (35)$$

with $U(\theta)$ being defined in (16) and ξ being a positive constant. By differentiating (35) with respect to time and substituting $\boldsymbol{\omega} = \boldsymbol{\omega}_{\text{ref}}$ given by (34), (17) (for $c(t) = v_d^2$), the rotational constraints (9), (11), and (7) or (12) we get:

$$\dot{W} = -\alpha_\tau^2 \|\tilde{\boldsymbol{\tau}}\|^2 - \beta_\tau^2 \|\mathcal{I}(\tilde{\boldsymbol{\tau}})\|^2 - \boldsymbol{\omega}_\tau^\top \mathbf{d}\left(\tfrac{\cos\theta - 1}{\cos\theta}\right) v_d \quad (36)$$

$$- \xi v_d^2 \tan^2\theta + \tilde{\mathbf{d}}\left(\frac{1}{\gamma_d}\dot{\tilde{\mathbf{d}}} + v_d \boldsymbol{\omega}_\tau\right) \quad (37)$$

In order to cancel the last term of the right side part of (36) we set $\dot{\tilde{\mathbf{d}}} = -\gamma_d v_d \boldsymbol{\omega}_\tau$ which corresponds to the update law (32). By using (32) and the inequality:

$$\boldsymbol{\omega}_\tau^\top \mathbf{d}\left(\frac{\cos\theta - 1}{\cos\theta}\right) v_d \leq \frac{\|\boldsymbol{\omega}_\tau\|^2}{4} + \|\mathbf{d}\|^2 v_d^2 \left(\tfrac{\cos\theta - 1}{\cos\theta}\right)^2 \quad (38)$$

we can upper-bound $\dot{W}$ (36) as follows:

$$\dot{W} \leq -\alpha_\tau^2 \|\tilde{\boldsymbol{\tau}}\|^2 - \beta_\tau^2 \|\mathcal{I}(\tilde{\boldsymbol{\tau}})\|^2 + \frac{\|\boldsymbol{\omega}_\tau\|^2}{4} \quad (39)$$

$$- \xi v_d^2 \tan^2\theta + \|\mathbf{d}\|^2 v_d^2 \left(\tfrac{\cos\theta - 1}{\cos\theta}\right)^2 \quad (40)$$

By expanding $\|\boldsymbol{\omega}_\tau\|^2$, using (33), setting $\xi > \|\mathbf{d}\|^2$ and after some trigonometric calculations we get:

$$\dot{W} \leq -\tfrac{\alpha_\tau^2}{4}\|\tilde{\boldsymbol{\tau}}\|^2 - \tfrac{\beta_\tau^2}{4}\|\mathcal{I}(\tilde{\boldsymbol{\tau}})\|^2 - \|\mathbf{d}\|^2 v_d^2 \left(\tfrac{1 - \cos\theta}{\cos\theta^2}\right) \quad (41)$$

Since $\cos\theta \leq 1$ and $\theta(t) \in D$ provided that $\theta(0) \in D$ (Theorem 1), the derivative of function W (35) can be upper-bounded as follows:

$$\dot{W} \leq -\tfrac{\alpha_\tau^2}{4}\|\tilde{\boldsymbol{\tau}}\|^2 - \tfrac{\beta_\tau^2}{4}\|\mathcal{I}(\tilde{\boldsymbol{\tau}})\|^2$$

Hence, $W(t) \leq W(0)$, $\forall\, t$ which implies that $\mathcal{I}(\tilde{\boldsymbol{\tau}})$ and $\tilde{\mathbf{d}}$ are bounded. Consequently, by using (34) and taking into account the constraints (9), (11), and (7) or (12), $\tilde{\boldsymbol{\tau}}$ is bounded and hence $\dot{\widehat{\mathbf{d}}}$ is bounded. Using the aforementioned boundedness results as well as those implied by Theorem 1, it can be easily proved by differentiating $\dot{W}$ that $\ddot{W}$ is bounded. Hence, applying Barbalat's Lemma we get that $\tilde{\boldsymbol{\tau}} \to 0$, $\mathcal{I}(\tilde{\boldsymbol{\tau}}) \to 0$. Using the aforementioned convergence results as well as $\theta \to 0$, it can be shown that $\widehat{\mathbf{d}} \to \mathbf{d}$ provided that v_d satisfies the persistent excitation condition and hence $\boldsymbol{\omega} \to \mathbf{d} v_d$. ■

C. Summary and Discussion

The proposed controller produces local estimates of the unconstrained motion direction and axis of rotation (in case of rotational door) using the update laws (25) and (32) respectively. The estimates are used within velocity references (19), (31) that enforce the robot to move with a desired velocity while controlling forces/torques along the constrained directions to small values guaranteeing compliant behavior.

The main condition for guaranteed performance is that the initial estimate is not perpendicular to the true value i.e. $\theta(0) \in D$. A typical example where this condition is not satisfied could be when opening a drawer with an initial estimate that it is sliding door (c.f. Fig. 2, cases 4 and 5). This issue can be overcome by using a moderate deviation in the initial estimate (see Section IV). The proposed method alone can not handle the case where the initial estimate is in the opposite direction of the true value, as this would generate a closing motion. This can be handled by an external monitoring system that stops the motion and retries with a different initial estimate if measured forces are too high, similar to a human who first pushes a door, and when it doesn't open, tries to pull it instead.

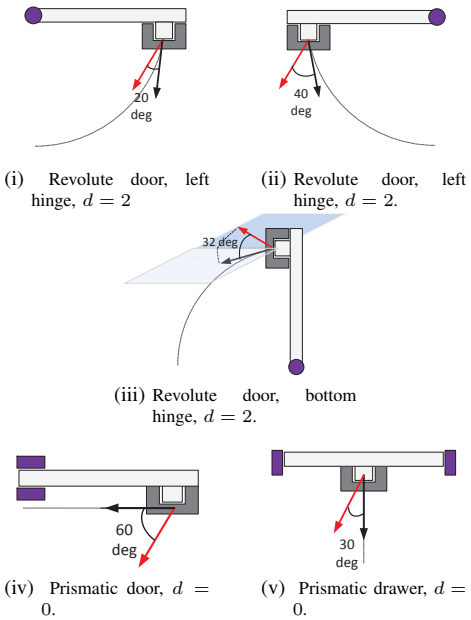
By defining the controller in the end-effector frame and estimating the motion directions locally, the proposed method is applicable to both revolute and prismatic doors/drawers. Coupling the estimator with the controller makes the method inherently on-line, and enables proofs of both the convergence of estimated parameters to true values and convergence of force/torque errors.

It is trivial to extend the method to produce explicit estimates of the physical location of the hinge of a revolute door, as estimates of both radial direction and radial distance are available. If we make the assumption that a large enough radial distance (we arbitrarily choose 10 m) implies a prismatic mechanism, the following steps will identify the hinge position: *Step 1)*: Assume a rotational mechanism if $\|\widehat{\mathbf{d}}\| > 0.1$. For $\|\widehat{\mathbf{d}}\| < 0.1$ we assume a sliding door or a drawer and do not proceed further. *Step 2)*: Calculate the estimated radial direction, coinciding with handle frame axis ${}^e\widehat{\underline{\mathbf{y}}}_h$, by the outer product of ${}^e\widehat{\underline{\mathbf{x}}}_h$, $\widehat{\mathbf{d}}\|\widehat{\mathbf{d}}\|^{-1}$. *Step 3)*: Center of rotation $\widehat{\mathbf{p}}_o$ is then: $\widehat{\mathbf{p}}_o = \mathbf{p}_e + \mathbf{R}_e {}^e\widehat{\underline{\mathbf{y}}}_h \|\widehat{\mathbf{d}}\|^{-1}$.

Note that the proposed control scheme can be directly implemented on any velocity-controlled robotic manipulator with a force/torque sensor in the end-effector frame. Implementation on a torque controlled robot may require the reference acceleration, i.e. time derivative of reference velocity. Then, force/torque feedback terms $\mathbf{v}_f$, $\boldsymbol{\omega}_\tau$ should only consist of the integral of the force error (projected on the constrained direction) and the torque, so that differentiation of noisy force/torque measurements is avoided.

IV. Scenarios and Evaluation

To demonstrate the generality of the approach, we consider five different scenarios covering five common cases found in domestic environments, see Fig. 2. All cases are treated with the same initial estimates and controller gains. Cases (i) and (ii) are typical revolute doors with vertical axis, with the hinge to the left or to the right, respectively. Case (iii) models a revolute door with axis of rotation parallel to the floor, such as is common for ovens. The radius of these door are all set to 50 cm. Case (iv) models a sliding door, and case (v) a typical drawer. The common initial estimate used in all cases is that of a prismatic joint, assuming $\hat{\mathbf{d}}(0) = \mathbf{0}$. The initial estimate of the unconstrained direction of motion is 30° offset from the normal direction to the plane of the door or drawer. The initial estimates are shown as red arrows, and the true direction is shown as black arrows in Fig. 2. The angular values given are the initial errors of the estimates. The controller gains are chosen as follows: $\alpha_f = \alpha_T = 0.05$, $\beta_F = \beta_T = 0.005$, $\gamma = \gamma_d = 2000$. The desired motion velocity is 5 cm/s, given as $v_d = 5(1 - e^{-10t})$ cm/s to avoid sharp initial transients.



(i) Revolute door, left hinge, $d = 2$

(ii) Revolute door, left hinge, $d = 2$.

(iii) Revolute door, bottom hinge, $d = 2$.

(iv) Prismatic door, $d = 0$.

(v) Prismatic drawer, $d = 0$.

Fig. 2 : The five different simulation cases. The angular measurement indicates the initial error. Note that all scenarios are initialized with the same estimate.

In Fig. 3 - upper plot, the response of the motion axis estimation errors are shown; convergence to the actual axis is achieved even for big initial errors. Figure 3 - lower plot depicts the estimation of the inverse signed distance d between the end-effector and the hinge; note that distance parameter estimate $\widehat{d}$ is not modified when the estimate coincides with the actual parameter and it converges to its actual value in all cases. Combining estimates of the modulated rotation axis with the motion axis we can calculate the center of rotation of the rotational doors in real time; simulation gives errors approximately 1.4 cm after 1.5 sec, or opening the door 7.5 cm. Given the threshold of $\|\widehat{\mathbf{d}}\| > 0.1$, the revolute doors are identified as such after 0.2 s. Fig. 4 shows the responses of the Euclidian norms of force and torque errors ($e_f = \|\bar{\mathbf{P}}({}^e\underline{\mathbf{x}}_h)\tilde{\mathbf{f}}\|$ and $e_\tau = \|\tilde{\boldsymbol{\tau}}\|$ respectively). Errors converge to zero following the convergence rate of

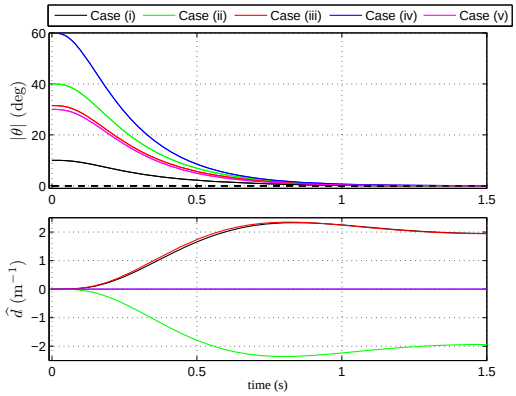


Fig. 3 : Estimation responses, upper figure: estimation error response in the orientation of motion axis, lower figure: response of the inverse distance between hinge and end-effector. Note that the true values are 2 for cases (i) and (iii), -2 for case (ii), and 0 for cases (iv) and (v).

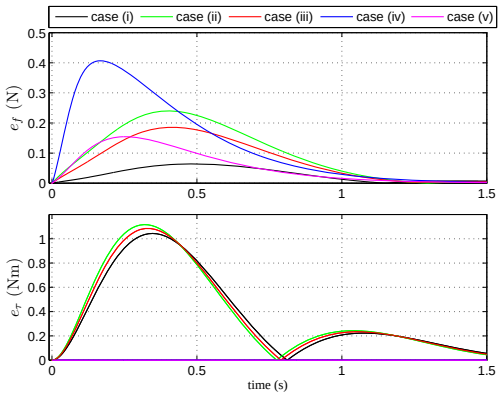


Fig. 4 : Force and torque responses, upper figure: norm of the projected force error, lower figure: norm of the torque error

modulated rotation axis and motion axis.

V. Conclusions

We proposed a unified method for manipulating different types of revolute and prismatic mechanisms. The method is model-free and it can be used to identify the type and the geometrical characteristics of one-joint mechanisms. By coupling estimation and action the method is inherently online and can be used in real-time applications. The method consists of a generalized velocity controller using estimates of the local motion direction, the axis of rotation and update laws for the estimated vectors. The design of the overall scheme guarantees compliant behavior and convergence of the estimated vectors to their actual values. Current work includes the integration of the proposed method with the vision based handle and door detection as well as long term experiments in domestic environments.

Acknowledgment

This work has been supported by the Swedish Research Council (VR) and Swedish Foundation for Strategic Research (SSF).

References

[1] K. Nagatani and S. Yuta, "An experiment on opening-door-behavior by an autonomous mobile robot with a manipulator," in *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems 95*, vol. 2, aug 1995, pp. 45 –50.

[2] G. Niemeyer and J.-J. Slotine, "A simple strategy for opening an unknown door," in *1997 IEEE International Conference on Robotics and Automation*, vol. 2, apr 1997, pp. 1448 –1453 vol.2.

[3] E. Lutscher, M. Lawitzky, G. Cheng, and S. Hirche, "A control strategy for operating unknown constrained mechanisms," in *IEEE International Conference on Robotics and Automation*, may 2010, pp. 819 –824.

[4] D. Ma, H. Wang, and W. Chen, "Unknown constrained mechanisms operation based on dynamic hybrid compliance control," in *IEEE International Conference on Robotics and Biomimetics*, dec. 2011, pp. 2366 – 2371.

[5] L. Peterson, D. Austin, and D. Kragic, "High-level control of a mobile manipulator for door opening," in *2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2000, pp. 2333 –2338.

[6] A. Jain and C. Kemp, "Pulling open novel doors and drawers with equilibrium point control," in *9th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2009*, dec. 2009, pp. 498 –505.

[7] ——, "Pulling open doors and drawers: Coordinating an omni-directional base and a compliant arm with equilibrium point control," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 1807–1814.

[8] M. Prats, S. Wieland, T. Asfour, A. del Pobil, and R. Dillmann, "Compliant interaction in household environments by the Armar-III humanoid robot," in *8th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2008*, dec. 2008, pp. 475 –480.

[9] J. Sturm, C. Stachniss, and W. Burgard, "A probabilistic framework for learning kinematic models of articulated objects," *Journal of Artificial Intelligence Research*, vol. 41, pp. 477–526, 2011.

[10] A. Petrovskaya and A. Y. Ng, "Probabilistic mobile manipulation in dynamic environments with application to opening doors," in *Proc. of the 20th International Joint Conference on Artifical Intelligence*, Hydrabad, India, January 2007, pp. 2178–2184.

[11] C. Ott, B. Bäuml, C. Borst, and G. Hirzinger, "Employing cartesian impedance control for the opening of a door: A case study in mobile manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on mobile manipulators: Basic techniques, new trends & applications*, 2005.

[12] C. Kessens, J. Rice, D. Smith, S. Biggs, and R. Garcia, "Utilizing compliance to manipulate doors with unmodeled constraints," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, oct. 2010, pp. 483 –489.

[13] W. Chung, C. Rhee, Y. Shim, H. Lee, and S. Park, "Door-opening control of a service robot using the multifingered robot hand," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 3975–3984, oct. 2009.

[14] H. Arisumi, J.-R. Chardonnet, and K. Yokoi, "Whole-body motion of a humanoid robot for passing through a door - opening a door by impulsive force -," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, oct. 2009, pp. 428 –434.

[15] Y. Karayiannidis, C. Smith, F. V. na, P. Ögren, and D. Kragic, "'open sesame!' - adaptive force/velocity control for opening unknown doors," in *IEEE/RAS Conference on Intelligent Robots and Systems*, Vilamoura, Portugal, Oct 2012.

[16] Y. Karayiannidis, C. Smith, P. Ögren, and D. Kragic, "Adaptive force/velocity control for opening unknown doors," in *IFAC Symposium on Robot Control*, Dubrovnik, Croatia, Sep 2012.