Robust Precision Manipulation With Simple Process Models Using Visual Servoing Techniques With Disturbance Rejection

Berk Calli, *Member, IEEE*, and Aaron M. Dollar, *Senior Member, IEEE*

***Abstract*— This paper presents a high-performance vision-based precision manipulation technique that does not rely on an object, contact, or gripper model, which are challenging and often times impractical to acquire. Instead, we utilize a simple process model that roughly maps object velocities to actuator velocities, and we maintain system efficiency and robustness via advanced vision-based control techniques with disturbance rejection mechanisms. For obtaining simple models, we derive a set of actuator coordination rules for achieving common task space motions. The performance degradation due to modeling inaccuracies is then minimized via the model predictive control framework and a correction matrix method. Our experimental results show that the proposed strategy results in high-performance precision manipulation with minimal modeling effort.**

***Note to Practitioners*—Compliant, soft robotic grippers make it easier to grasp objects with various shapes and sizes; these grippers adapt to the shape of the object, which provides robustness to positioning errors and often removes the necessity to precisely plan the contact locations. These advantages make compliant grippers ideal to use in industrial settings as well as in service robotics, where the variety of object shapes and sizes are immense. On the other hand, for the tasks that require precise object manipulation (e.g., for a peg-in-hole problem), these hands are more challenging to control than their rigid counterparts: it is harder to obtain their precise models, and they often do not have enough proprioceptive sensors to calculate the full pose of the system. In this paper, we propose solutions to utilize vision feedback for positioning an object using compliant hands. These solutions do not rely on precise models of the gripper or the full knowledge of the gripper state. We adopt various control techniques to provide precise positioning in steady state as well as to maintain efficiency in the transient.**

***Index Terms*— Dexterous manipulation, in-hand manipulation, model predictive control (MPC), visual servoing.**

I. Introduction

THE ability to conduct in-hand precision manipulation adds dexterity to a robotic manipulator; the additional mobility supplied by the robot's gripper helps the system to work around obstacles and avoid joint singularities [1]. Moreover, lower finger inertias compared to the inertia of the full arm allow energy efficient and precise positioning of the target object. These features can be considered as building blocks of a human-like dexterity, which is especially needed for home/service robots.

In general, manipulation phenomena in robotics are challenging to model, and in-hand manipulation is no exception: for obtaining a reliable process model of an in-hand manipulation task, one needs to have an accurate gripper model, contact model, object model, and knowledge of the contact locations [2]–[5]. Assuming this information is available along with the sensors that can measure the necessary system states, a model-based strategy can be employed for planning an in-hand manipulation task by calculating joint positions and contact forces. Nevertheless, such an approach poses practical challenges as each of these models are hard to acquire precisely, especially in uncontrolled environments, and any modeling imprecision reduces accuracy and robustness while increasing the risk of task failure, i.e., dropping the object. Moreover, sensors necessary for measuring the required system states (i.e., joint encoders and force sensors) complicate the hand designs.

Our approach to in-hand manipulation combines advanced vision-based control techniques and system compliance to allow working with very rough process models while maintaining robustness and task execution efficiency. We derive simple models that approximate actuator inputs for a set of common (and often orthogonal) motions in task space for the purpose of simplifying the modeling and planning steps. Using these models with traditional image feature-based visual servoing techniques [6], [7], which are already known to be robust to modeling errors, can provide convergence in steady state as we have shown in our previous work [8]. However, significant performance degradation and robustness issues can be observed in transient response when the error between the derived model and the actual system is large (an example can be seen from Fig. 1). Such performance degradation can be crucial for a precision manipulation task, where the expected/required system accuracy is usually high. Rather than using traditional visual servoing schemes, the effect of modeling inaccuracies can be minimized using algorithms with disturbance rejection; in this paper, we use a vision-based model predictive control (MPC) algorithm and a correction

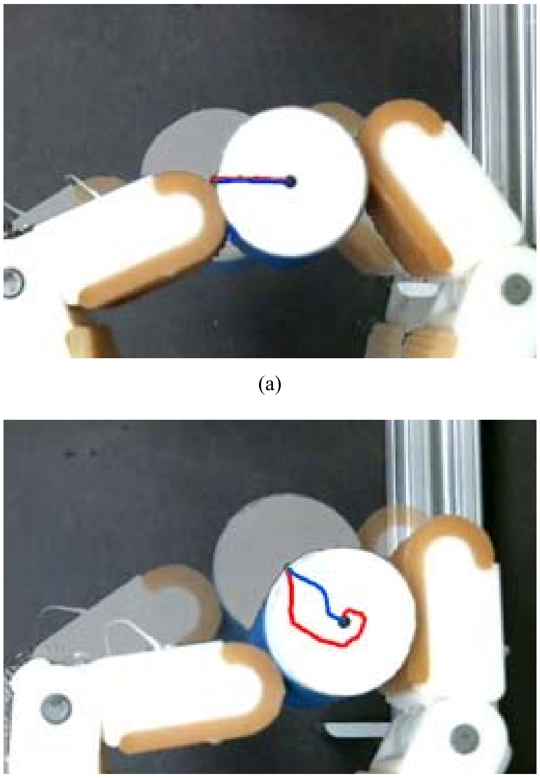
Manuscript received January 20, 2018; accepted March 15, 2018. This paper was recommended for publication by Associate Editor H. Liu and Editor H. Ding upon evaluation of the reviewers' comments. This work was supported by the U.S. National Science Foundation under Grant IIS-1317976. *(Corresponding author: Berk Calli.)*

The authors are with the Department of Mechanical Engineering and Materials Science, Yale University, New Haven, CT 06511 USA (e-mail: berk.calli@yale.edu; aaron.dollar@yale.edu).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. The Supplementary Material contains one video that explains the methodology of the paper, advantages of the presented approach and videos of some experimental results. This material is 52.5 MB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2018.2819661



(a)

(b)

Fig. 1. Vision-based in-hand manipulation using simple models with (blue line) and without (red line) MPC. (a) Region of the workspace, where the simple models represent the system accurately. (b) In particular, difficult part of the workspace where object roles over the fingertip, and the models get inaccurate. The same model parameters and control gains are used for both executions. It can be seen that even without MPC, the simple models coupled with vision feedback can successfully meet the references in steady state, but inefficiencies are observed when model inaccuracies are high. MPC strategy helps to recover efficiency and reduces the risk of task failure.

matrix method. In this way, robustness and system efficiency can be maintained even with very rough process models. Moreover, the influence of unmodelled phenomena (e.g., friction and sliding) on the system performance can be reduced.

Our approach is an excellent choice, especially for precision manipulation with adaptive/underactuated grippers [9]–[12]. These grippers are hard to model due to their elastic elements, and estimating their adaptive behavior requires the knowledge of the contact locations and force magnitudes. Acquiring this information either requires a dense array of force sensors or multiple joint encoders, which complicate their mechanical design that was aimed to be simple and effective in the first place. On the other hand, adaptive/underactuated grippers mechanically provide a substantial amount of compliance, which helps to maintain robust object–gripper interaction during manipulation. With our approach, simplified models of these systems can be utilized, and compliance and advanced visual servoing algorithms provide the robustness and efficiency without the need for joint position or force measurements. In this paper, we use underactuated grippers such as the Model T42 and Model M2 (modified for actuating the rigid finger) as our test beds [13]. Nevertheless, our method can also be utilized for fully actuated grippers since similar compliant behavior can be implemented using the methods like [14] and [15], and in this way, the abovementioned modeling and planning difficulties can similarly be overcome.

This paper extends our previous work [16] with added discussions, examples, and experiments: we provide additional gripper-specific models, provide a comprehensive discussion of the in-hand manipulation literature and novelty of our method therein, experimentally show the accuracy of the models over the manipulation workspace, and analyze the advantages of the MPC framework when modeling inaccuracies are large, and present an analysis of the effect of MPC parameters on the system performance. This paper is organized as follows. The relating literature of in-hand precision manipulation is provided in Section II. Modeling stage is explained for two types of grippers in Section III. The use of the derived models within a traditional visual servoing scheme, with the correction matrix method and the MPC-based visual servoing method are given in Section IV. The experimental results are presented in Section V, and the conclusions are drawn in Section VI.

II. Related Work

This section covers the in-hand manipulation strategies in the literature, and presents a discussion of the primitives/synergies ideas in the trajectory generation and grasping literature along with their resemblance with our method. Following that we present methods that utilize vision feedback in in-hand precision manipulation and the use of MPC framework in robotics applications.

A. In-Hand Manipulation

In the literature, various closed-form in-hand manipulation models are presented depending on the contact model assumption: the analysis in [17] assumes stationary point contacts with no rolling or sliding; methods in [2], [3], and [18] assume rolling point contacts while neglecting the sliding effect; and a review on soft-contact modeling can be found in [19]. These models are crucial to understand the in-hand manipulation phenomena and its challenges. However, they are impractical for manipulation planning due to the difficulties in acquiring the necessary submodels, the sensor requirements and the planning complexity.

1) Modeling Difficulties: The process model of an in-hand manipulation task is composed of an object model, a contact model, and a gripper model, and obtaining each of them accurately can be challenging in many common scenarios. In unstructured environments, object models are not often available *a priori*. Even though there are methods in the literature that do not rely on an object model for in-hand manipulation [20], [21], they require a specific gripper topology. For incorporating the contact models, the friction coefficients between the object and the fingers need to be known, along with the surface curvature at contact locations. Even if this information is available, calculating sliding effects, which occur frequently during in-hand manipulation, requires switching between models, and introduces additional complexity to planning [22]. Regarding the grippers, reliable models may be available via manufacturer for rigid types. However, for adaptive/underactuated grippers, the models require the knowledge of elastic element characteristics [18], which can be highly nonlinear, hard to acquire, and may change over time. While composing an overall process model, inaccuracies in these submodels accumulate and cause significant performance degradation both in transient and steady state, or may even result in failure of the task, i.e., dropping the target object.

2) Required Sensors: Model-based techniques require the state of the system to be measured, i.e., gripper joint positions, contact locations, and/or contact forces. Acquiring this information necessitates sensors such as joint encoders and force/torque sensors, which complicate the hand design and hamper its compactness.

3) Planning Complexity: Planning an in-hand manipulation task, that is calculating joint locations throughout the trajectory using the models, can be complicated depending on the type of the gripper. For redundantly actuated hands, the redundancy needs to be handled [23], whereas for underactuated hands, since each joint cannot be controlled independently, the finger trajectories need to be estimated by taking the adaptive behavior into account.

Our strategy is to avoid these difficulties by adopting simple modeling procedures, while handling inaccuracies using advanced vision-based methods and system compliance. In our applications, this strategy also eliminates the need for joint encoders and force sensors.

B. Primitives/Synergies and the Effect of Compliance

Handling modeling and planning complexity is also required in other robotics problems such as grasping. Grasp planning is a complicated procedure if the problem is formulated as deciding how to position the robotic fingers on a target object precisely for achieving a stable grasp. Instead, inspired from neuroscience, researchers develop the concepts of grasping primitives [24], eigengrasps [25], [26], and synergies [27], [28]. In these concepts, dominantly used motions in task space are detected, and actuator inputs that generate these motions are determined. Consecutively, it is shown that a large set of grasping abilities can successfully be executed using a small number of task space-actuator space relations. Therefore, grasp planning is simplified to choosing the right (or a combination of) primitives/synergies for a given target object and scenario. Nevertheless, for these methods to be generalized and be successful for a large variety of object shapes and sizes, the system needs to have a degree of compliance [29]. This can be achieved either mechanically (e.g., as in adaptive grippers [9], [10], and [27]), or algorithmically [14], [15], and [27].

We apply a similar strategy to the in-hand precision manipulation problem: we determine actuator inputs that generate commonly used object motions in Cartesian space. By using these relations, we obtain a rough model between the actuator inputs and the object motion. Similar to grasping primitives/synergies, we also rely on the system compliance for keeping the contact with the object during manipulation. However, different from the grasping case, our system requires high precision: the success of a grasp is often measured in a binary sense, i.e., dropping or not dropping the target object. In that case, compliance provides a large enough error margin to keep the object intact with the hand. Consequently, imprecise motions of the fingers in grasping are tolerable. In the case of in-hand precision manipulation, we do not only need to maintain the contact with the object, but also to move the object with high accuracy. The discrepancy between the actual system model and the rough models can generate significant errors in task space both in the transient and the steady state. By using vision feedback in traditional schemes [6], [7], convergence can be achieved in steady state [8], while transient performance is still affected by modeling errors. Robustness issues may also be observed if the errors are high. We, therefore, design visual servoing algorithms with disturbance rejection mechanisms to minimize these modeling error effects. In Section II-C, the literature on in-hand manipulation using vision feedback is covered.

C. Vision-Based In-Hand Manipulation

In the literature, visual servoing [30] is used for robotic manipulation [31], control of mobile robots [32], and microrobotics [33] among many other applications. The use of vision feedback is preferable for in-hand manipulation since it allows closing the loop in task space without complicating the hand design. Nonetheless, these sensors are also needed to perceive the state of the environment for many common manipulation tasks that require relative positioning (e.g., peg-in-hole and key insertion).

In [34], a vision-based control method is proposed within the optimal control framework. By utilizing force sensors, this method can also control the amount of applied force to the object. A sensor fusion approach is proposed in [35], in which vision feedback, force sensing, and joint feedback are used to deal with the sliding motion and external disturbances. Both of these methods rely on accurate hand and object models. The experimental evaluation of these algorithms in the literature is limited to a few case studies.

With a similar motivation to our approach, an adaptive visual servoing algorithm is utilized in [36] and [37] for reducing the dependence on accurate models. This algorithm updates the visual-motor Jacobian during the execution by minimizing the error between the observed and expected velocities. A trust-region strategy is also employed to keep the controller within the validity region of the Jacobian. If a good initial model is available, this method is excellent to handle the changes in Jacobian matrix due to contact location changes. However, for the models as rough as the ones obtained in this paper, large errors between the real and estimated Jacobians may cause undesired, shaky motions in the transient, which cause unstability. Moreover, these types of adaptive schemes are sensitive to control lag and rely on the performance of the lower level controller, since they operate in the kinematics level. Due to these problems, we were not able to achieve a stable precision manipulation performance using our simple models with the proposed adaptive schemes in the literature.

D. Model Predictive Control

One of the methods we use to minimize the effect of modeling errors to the system performance is MPC [38]. Typically, in the MPC framework, the control signal is obtained by solving an optimization problem over a finite horizon in each control step of the execution. In robotics, MPC is used in vision-based mobile robot navigation [39]–[41], hybrid position/force control [42], bipedal locomotion [43], vision-based control of underwater vehicles [44], robotic heart surgery [45], [46], and recently in the control of soft robots [47].

The popularity of the MPC framework in robotics lies in its ability to handle constraints and disturbances: in MPC, kinematics, dynamics, and workspace constraints of a system can easily be incorporated to the optimization problem. Similarly, the effect of disturbances can be estimated and minimized by using a disturbance model. In the simplest case, this model assumes that the effect of disturbances will remain the same with the previous step in the finite horizon. If a more accurate model is available, a better disturbance rejection performance can be achieved (e.g., the current disturbance model for underwater vehicles in [44]). Alternatively, if the effect of disturbances is known from the previous executions of the task, these effects can be integrated to the optimization problem directly (e.g., the disturbance caused by periodic heartbeat and inspiration in heart surgery [46]). As a combination of these two strategies, past experience can be modeled and integrated to the optimization problem (e.g., the mobile robot using past experiences to model terrain disturbances [40]). In our work, we consider modeling errors as disturbances, and minimize their effect using the MPC framework. We also utilize and compare periodic and nonperiodic disturbance models for repeating references.

III. Obtaining Simple Motion Models

As explained in Section II-A, obtaining an accurate motion model for the in-hand manipulation process is a challenging and requires information such as object model, friction coefficients, and contact locations, which are unavailable in many manipulation scenarios. Instead our strategy is to utilize simple models by actuator synchronization rules for common motions in the task space; with the help of system compliance and advanced visual servoing techniques, we show that high performance precision manipulation can be realized even with these rough models without any joint position feedback, force feedback, or sophisticated planning schemes.

In this section, we explain how to derive the simple models for two grippers designed in our laboratory, Model T42 and Model M2, and then present a discussion on its generalization. It is assumed that an initial stable grasp is maintained *a priori*.

A. Model T42 Gripper

Model T42 [Fig. 2(a) and (b)] has two identical opposing fingers that have two joints and one actuator. The gripper provides mechanical compliance during a precision grasp; when the springs are active (not in the resting state), they provide restoring force that keeps the contact with the object.

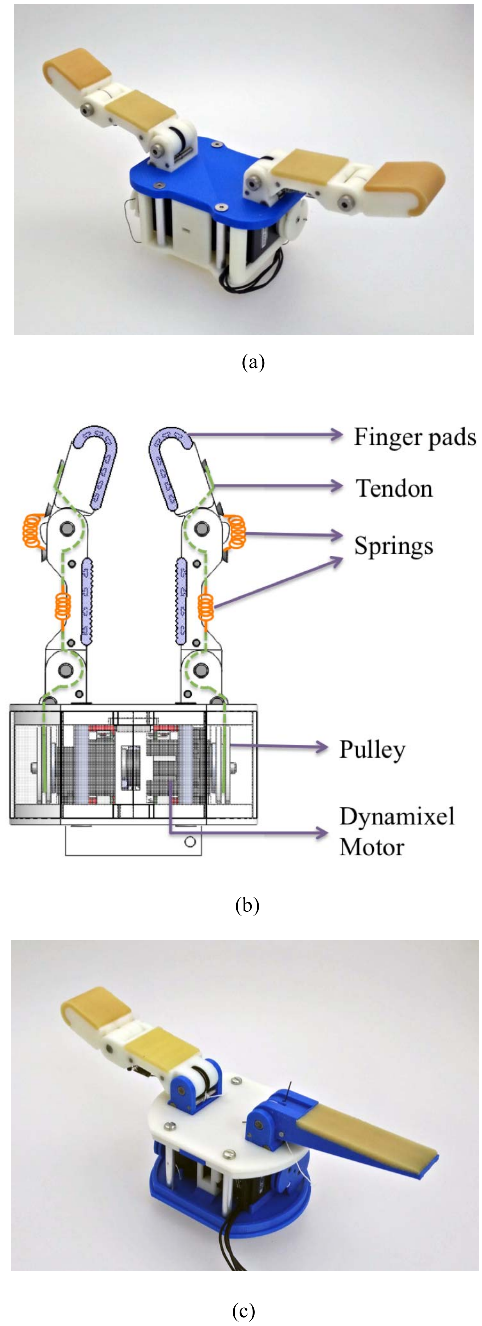


Fig. 2. Grippers used in our experiments. (a) Model T42. (b) Detailed schema for Model T42. (c) Modified Model M2 with actuated rigid finger.

This gripper is capable of planar precision manipulation, and since it has two degrees of freedom, the object pose cannot be simultaneously controlled in all three dimensions of the planar Cartesian workspace (position in the x-direction, position in the y-direction, and orientation around the manipulation plane), but in its 2-D submanifold.

We generate a rough Jacobian that relates Cartesian space motion of the object and the actuator velocities with the following simple observations that are also as depicted in Fig. 3. Moving the actuators to opposite directions by the same amount moves the object along the x-direction while rotating it; moving the object in the negative x-direction rotates the

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

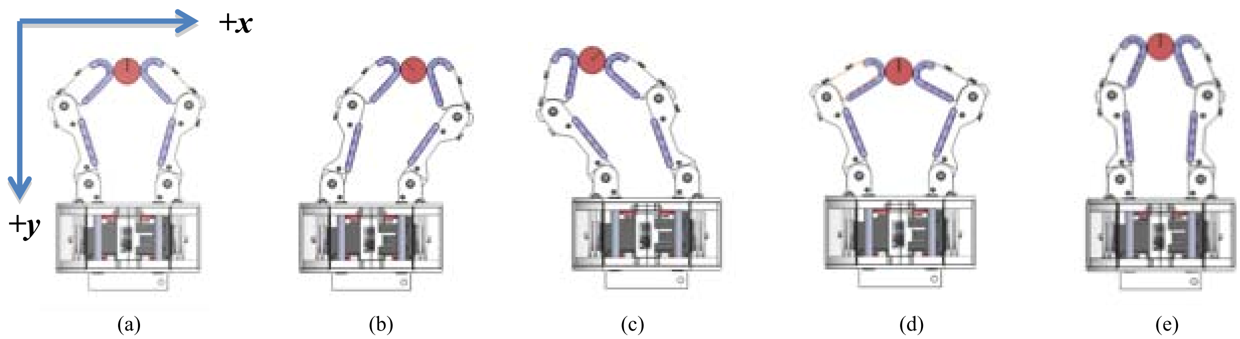


Fig. 3. Manipulation with Model T42 hand. (a) Initial grasp configuration, (b) and (c) when the motors are moved in the opposite direction with the same amount the object moves left or right (depending on the direction), and (d) and (e) when the motors are moved in the same direction with the same amount, the object is moved up or down (depending on the direction).

object clockwise and vice versa [Fig. 3(b) and (c)]. If we neglect the motion in the y-direction, these relations between the object motion and actuator velocities are expressed as follows:

$$V_{\text{ox}} = K_x\dot{q}_1 = -K_x\dot{q}_2 \tag{1}$$
$$V_{\text{oy}} = 0 \tag{2}$$
$$V_{o\theta} = -K_{\theta x}\dot{q}_1 = K_{\theta x}\dot{q}_2. \tag{3}$$

Here, V_{ox} and V_{oy} are the linear velocities in the x- and y-directions, $V_{o\theta}$ is the angular velocity around the manipulation plane normal, q_1 and q_2 are actuator positions, and K_x and $K_{\theta x}$ are scalars.

Our second observation is that moving the actuators to the same direction in the same amount moves the object along the y-direction while rotating it [Fig. 3(d) and (e)]. The rotation direction depends on whether the object is at the right or left side of the gripper's symmetry axis. If the object is at the right side, moving the object to the $+y$-direction rotates it counterclockwise and vice versa. The amount of rotation is related to the distance of the object to the symmetry axis: no rotation is observed on the symmetry axis, and the amount of rotation increases by going further than the axis. If the motion along the x-direction is neglected, these relations are expressed as follows:

$$V_{\text{ox}} = 0 \tag{4}$$
$$V_{\text{oy}} = K_y\dot{q}_1 = K_y\dot{q}_2 \tag{5}$$
$$V_{o\theta} = -pK_{\theta y}\dot{q}_1 = -pK_{\theta y}\dot{q}_2. \tag{6}$$

Here, p is the distance of the object to the symmetry axis and K_y and $K_{\theta y}$ are constant scalars.

As mentioned earlier in this section, since Model T42 has two degrees of freedom, it can only control the object pose within the 2-D manifold of the 3-D workspace (if controlled sliding can be applied on the object, this 2-D manifold can be altered; this aspect is out of the scope of this paper, but will be our future work). By combining (1)–(6), we have derived the following three Jacobians to be used in the visual servoing loop: for mapping the velocities in the x- and y-directions to the actuator velocities, the (1) and (5) are combined

$$\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\end{bmatrix} = \boldsymbol{J}_{\boldsymbol{s}_{x,y}}\begin{bmatrix}V_{\text{ox}}\\ V_{\text{oy}}\end{bmatrix} = \begin{bmatrix}\dfrac{1}{K_x} & \dfrac{1}{K_y}\\ -\dfrac{1}{K_x} & \dfrac{1}{K_y}\end{bmatrix}\begin{bmatrix}V_{\text{ox}}\\ V_{\text{oy}}\end{bmatrix}. \tag{7}$$

For mapping translational velocity in the x-direction and rotational velocity around the manipulation plane, we combine (1) and (6) as follows:

$$\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\end{bmatrix} = \boldsymbol{J}_{\boldsymbol{s}_{x,\theta}}\begin{bmatrix}V_{\text{ox}}\\ V_{o\theta}\end{bmatrix} = \begin{bmatrix}\dfrac{1}{K_x} & -\dfrac{1}{pK_{\theta y}}\\ -\dfrac{1}{K_x} & -\dfrac{1}{pK_{\theta y}}\end{bmatrix}\begin{bmatrix}V_{\text{ox}}\\ V_{o\theta}\end{bmatrix}. \tag{8}$$

Similarly for translational velocity in the y-direction and rotational velocity around the manipulation plane, we combine (3) and (5)

$$\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\end{bmatrix} = \boldsymbol{J}_{\boldsymbol{s}_{y,\theta}}\begin{bmatrix}V_{\text{oy}}\\ V_{o\theta}\end{bmatrix} = \begin{bmatrix}\dfrac{1}{K_y} & -\dfrac{1}{K_{\theta x}}\\ \dfrac{1}{K_y} & \dfrac{1}{K_{\theta x}}\end{bmatrix}\begin{bmatrix}V_{\text{oy}}\\ V_{o\theta}\end{bmatrix}. \tag{9}$$

The accuracy of these Jacobians for representing the hand object system varies at different parts of the workspace. As can be seen from Fig. 4, the Jacobian in (7) represents the system more accurately at the center of the workspace, whereas the accuracy drops toward the boundaries. If no vision feedback is utilized, these inaccuracies can cause large positioning errors of the object. Vision feedback provides robustness to the inaccuracies by closing the loop in the task space, but if the modeling errors are large and are not handled explicitly with advanced control techniques, large deviations from the optimal path in the image space can be observed as also demonstrated in Fig. 1.

B. Model M2 Gripper

Model M2 is an asymmetric gripper with one underactuated finger identical to Model T42 and one flat actuated finger as can be seen from Fig. 2(c). For this gripper, a similar approach to the Model T42 case can be followed due to the similar topology. Nevertheless, Model M2 is specifically useful and

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

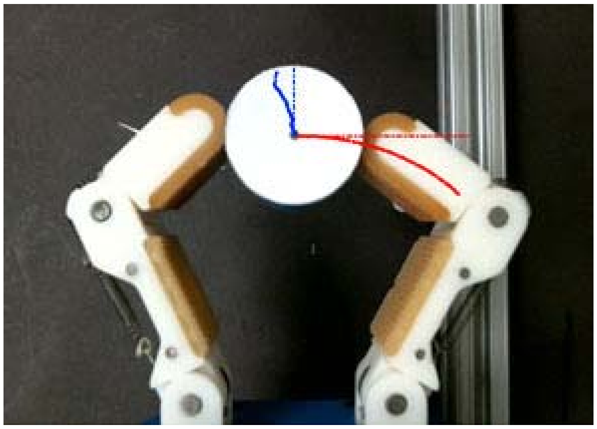


Fig. 4. Precision of the Jacobian in (7) designed for Model T42. Starting from the point at the center, pure horizontal (red lines), and vertical (blue lines) velocity signals are given. Dashed and solid lines represent ideal and actual trajectories, respectively. The accuracy of the Jacobian drops as the system moves toward the workspace boundaries.

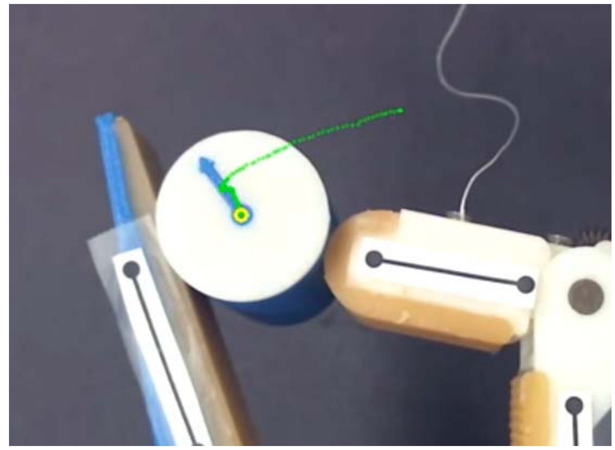


Fig. 5. In-hand manipulation with Model M2 gripper using the Jacobian obtained in Section III-B in the image-based visual servoing loop. The object follows a trajectory (green line) toward a reference point that is indicated with a yellow circle. The velocity vector is aligned with the flat finger direction, and the object is rolled on the flat finger toward the reference point.

designed for rolling/sliding the target object on the flat finger, and in order to exploit this property, we choose to design the Jacobian accordingly: in this design, the workspace is partitioned in such a way that the first column of the Jacobian is dedicated to align the velocity vector

$$\boldsymbol{V}_{\boldsymbol{o}_{\boldsymbol{xy}}} = \begin{bmatrix} V_{\text{ox}} \\ V_{\text{oy}} \end{bmatrix}$$

with the direction of the flat finger surface, and the second column is for sliding/roling the object along that finger as explained in detail as follows.

Let us call the angle of the velocity vector and the angle between the rigid finger and the gripper base β and ϕ, respectively. The angle ϕ can directly be derived from the actuator encoder since the finger is rigid, and β can be calculated as

$$\beta = a\tan2(V_{\text{oy}}, V_{\text{ox}}). \tag{10}$$

The error between these angles is defined as

$$e_{\alpha\phi} = \begin{cases} V_{\text{oy}} \geq 0 & \pi + \phi - \beta \\ V_{\text{oy}} < 0 & \phi - \beta. \end{cases} \tag{11}$$

Our aim is to make this error zero by moving the object along the x-axis so that the velocity vector is aligned with the direction of the flat finger surface. For this purpose, the velocity in the x-direction can be selected as

$$V_{\text{ox}} = -K_e e_{\alpha\phi} \tag{12}$$

where K_e is a positive scalar. Here, we use the same relation in (1), only by replacing V_{ox} with its value in (12)

$$e_{\alpha\phi} = -\frac{K_x}{K_e}\dot{q}_1 = \frac{K_x}{K_e}\dot{q}_2. \tag{13}$$

For the component of the velocity vector along the flat finger surface direction, we first calculate a rolling/sliding velocity V_s. This can be done by the inner product of the finger vector

$$\boldsymbol{F_f} = \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix}$$

and the velocity vector

$$V_s = \boldsymbol{V}_{\boldsymbol{o}_{\boldsymbol{xy}}}^T \boldsymbol{F_f}. \tag{14}$$

In order to slide/role the object by keeping the rigid finger steady and making the underactuated finger reconfigure against it, we have

$$V_s = K_s\dot{q}_1, \quad \dot{q}_2 = 0. \tag{15}$$

Using (13) and (15), we obtain the following Jacobian:

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} = \boldsymbol{J}_{\boldsymbol{s_{x,y}}} \begin{bmatrix} e \\ V_s \end{bmatrix} = \begin{bmatrix} -\dfrac{K_e}{K_x} & \frac{1}{K_s} \\ \dfrac{K_e}{K_x} & 0 \end{bmatrix} \begin{bmatrix} e \\ V_s \end{bmatrix}. \tag{16}$$

A trajectory obtained by using this Jacobian in visual servoing loop can be seen from Fig. 5, where the parameter K_e is set to a high value relative to K_s so that the object is aligned with the flat finger (to the sliding/rolling trajectory) quickly and then the rolling action takes over. Alternatively, these motions can be applied sequentially.

C. Notes on the Simple Models

In these two examples, the derivations of the simple models are based on simple intuitions about the gripper–object system (e.g., moving fingers to the left will make the object move left). For more complicated systems with higher degrees of freedom, these intuitions may be nontrivial to get. At this point, hand synergies framework [28] is instrumental to reduce the dimensionality of the gripper fingers for encompassing fundamental motions.

It is also crucial to maintain the grasp stability during the manipulation process. In underactuated/adaptive grippers, passive system compliance provides us a major advantage to maintain the stability by their elastic elements: the stability and manipulability analysis for underactuated systems in [48] states that any unconstrained motion of the hand object system requires an elastic element for restoring the contact in order to maintain stability. While deriving these models and using them in the control loop, this principle should also be taken into account, i.e., the elastic elements that help to keep contact with the object need to be kept loaded during the operation.

Assuming that the grasp stability is maintained as discussed above, objects with different shapes and sizes go through similar motions when the proposed actuation rules are applied. The main reason for this similar behavior is the compliance of the system. Compliance creates a basin that moves the system toward the minimum energy configuration, which is the case when the applied forces on the object are collinear. Therefore, contact frames that are used to construct the grasp matrix, and therefore, the structure of the matrix are very similar for objects with different shapes and sizes.

IV. Visual Servoing Using Simple Models

In this section, the use of the simple models with traditional image-based visual servoing scheme is presented, and two robust methods are proposed, namely, a correction matrix method and vision-based MPC.

A. Conventional Image-Based Visual Servoing

Generally, visual servoing schemes generate velocity references for the object using a proportional control rule

$$\boldsymbol{V}_{\boldsymbol{o}}^{\boldsymbol{Cam}} = -\boldsymbol{\lambda} \boldsymbol{J}_{\boldsymbol{int}}^{+} \boldsymbol{e}. \tag{17}$$

Here, $\boldsymbol{V}_{\boldsymbol{o}}^{\boldsymbol{Cam}}$ indicates the velocity reference for the object expressed in the camera frame, $\boldsymbol{e}$ is the feature error vector, $\boldsymbol{J}_{\boldsymbol{int}}$ is the interaction matrix, $\boldsymbol{J}_{\boldsymbol{int}}^{+}$ is its pseudo inverse, and $\boldsymbol{\lambda}$ is the diagonal gain matrix (in our 2-D implementation we use a point feature and the interaction matrix for point features, which is diagonal in 2-D [6]). Our error vector is the difference between the reference and current point locations in the x- and y-directions of the image space. Nevertheless, any other type of image features can be utilized in the presented framework. The resulting velocity reference needs to be transformed from the camera coordinate frame to the hand coordinate frame

$$\boldsymbol{V}_{\boldsymbol{o}}^{\boldsymbol{hand}} = \boldsymbol{J}_{\boldsymbol{Cam}}^{\boldsymbol{hand}} \boldsymbol{V}_{\boldsymbol{o}}^{\boldsymbol{Cam}}. \tag{18}$$

In order to project this velocity to the actuator space, we use inverse of the hand Jacobian $\boldsymbol{J}_{\boldsymbol{h}}^{-1}$ and the transpose of the grasp matrix $\boldsymbol{G}^{\boldsymbol{T}}$ [49] as follows:

$$\dot{q} = \boldsymbol{J}_{\boldsymbol{h}}^{-\mathbf{1}} \boldsymbol{G}^{\boldsymbol{T}} \boldsymbol{V}_{\boldsymbol{o}}^{\boldsymbol{hand}} \tag{19}$$

where

$$\dot{q} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}.$$

By combining the transformations in (17)–(20), we obtain a visual-motor Jacobian $\boldsymbol{J}$ that projects feature velocities in image space to actuator velocities

$$\boldsymbol{J} = \boldsymbol{J}_{\boldsymbol{h}}^{-\mathbf{1}} \boldsymbol{G}^{\boldsymbol{T}} \boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}} \boldsymbol{J}_{\boldsymbol{int}}^{+}. \tag{20}$$

In our framework, the derived Jacobians replace the $\boldsymbol{J}_{\boldsymbol{h}}^{-1} \boldsymbol{G}^{\boldsymbol{T}}$ part of the projection. If we consider the Model T42 hand, we need to choose one of the Jacobians in (7), (8), or (9), which are used to project V_o^{hand} to the actuator space

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_{\boldsymbol{s}} \boldsymbol{V}_{\boldsymbol{o}}^{\boldsymbol{hand}}. \tag{21}$$

Object velocity expressed in the hand frame should still be transferred to the camera frame to obtain a control rule for actuators

$$\dot{\boldsymbol{q}} = -\boldsymbol{\lambda} \boldsymbol{J}_{\boldsymbol{s}} \boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}} \boldsymbol{J}_{\boldsymbol{int}}^{+} \boldsymbol{e}. \tag{22}$$

By replacing $\boldsymbol{J}_{\boldsymbol{h}}^{-1} \boldsymbol{G}^{\boldsymbol{T}}$ with $\boldsymbol{J}_{\boldsymbol{s}}$, we remove the necessity of an object model, a contact model, and a detailed gripper model. The matrices $\boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}}$ and $\boldsymbol{J}_{\boldsymbol{int}}^{+}$ require the transformation between the camera and gripper frames and camera intrinsic parameters, respectively, both of which are often available in many robotics applications.

Of course $\boldsymbol{J}_{\boldsymbol{s}}$ is a rough approximation for $\boldsymbol{J}_{\boldsymbol{h}}^{-1} \boldsymbol{G}^{\boldsymbol{T}}$. However, visual servoing techniques provide robustness to these inaccuracies (including the errors in $\boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}}$ and $\boldsymbol{J}_{\boldsymbol{int}}^{+}$) and achieve convergence. Still, these inaccuracies affect the transient response of the system. To improve the transient response and robustness, we propose the following schemes.

B. Correction Matrix Method

The adaptive algorithms mentioned in Section II-C estimate the visual-motor Jacobian [$\boldsymbol{J}$ matrix in (20)] by iteratively minimizing the difference between the calculated and measured feature locations. These methods may lead to a loss of contact with the object while exploring the parameter space, since inaccuracies during the transient of the adaptation result in undesired, shaky motions. Moreover, within the framework of this paper, those adaptation schemes are not preferred as they alter the derived Jacobians, and gripper-specific characteristic motions cannot be maintained (e.g., the one designed for Model M2 in Section III-B).

Instead, we propose to calculate a projection matrix that maps the unit vector of current object velocity to the unit vector of the desired velocities

$$\bar{\boldsymbol{V}}_{\boldsymbol{o}}^{\boldsymbol{hand}^*} = \boldsymbol{H} \bar{\boldsymbol{V}}_{\boldsymbol{o}}^{\boldsymbol{hand}^{\boldsymbol{cur}}}. \tag{23}$$

In (23), $\bar{\boldsymbol{V}}_{\boldsymbol{o}}^{\boldsymbol{hand}^*}$ and $\bar{\boldsymbol{V}}_{\boldsymbol{o}}^{\boldsymbol{hand}^{\boldsymbol{cur}}}$ signify desired and current unit vectors, respectively, in the hand coordinate frame. In this formulation, it is preferred to calculate the projection between the unit vectors rather than the vectors with magnitudes in order to avoid velocity fluctuations and achieve smoother motions.

The unit vectors can be calculated by using desired and current image trajectories

$$\bar{\boldsymbol{V}}_{\boldsymbol{o}}^{\boldsymbol{hand}^*} = \boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}} \boldsymbol{J}_{\boldsymbol{int}}^{+} \boldsymbol{v}_{\boldsymbol{f}_{\boldsymbol{d}}} / \left| \boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}} \boldsymbol{J}_{\boldsymbol{int}}^{+} \boldsymbol{v}_{\boldsymbol{f}_{\boldsymbol{d}}} \right| \tag{24}$$

$$\bar{\boldsymbol{V}}_{\boldsymbol{o}}^{\boldsymbol{hand}^{\boldsymbol{cur}}} = \boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}} \boldsymbol{J}_{\boldsymbol{int}}^{+} \boldsymbol{v}_{\boldsymbol{f}} / \left| \boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}} \boldsymbol{J}_{\boldsymbol{int}}^{+} \boldsymbol{v}_{\boldsymbol{f}} \right|. \tag{25}$$

where $\boldsymbol{v}_{\boldsymbol{f}}$ and $\boldsymbol{v}_{\boldsymbol{f}_{\boldsymbol{d}}}$ are current and desired feature velocity vectors, respectively. The $\boldsymbol{H}$ matrix can then be used as a correction term in the projection

$$\dot{q} = -\boldsymbol{\lambda} \boldsymbol{J}_{\boldsymbol{s}} \boldsymbol{H} \boldsymbol{J}_{\boldsymbol{cam}}^{\boldsymbol{hand}} \boldsymbol{J}_{\boldsymbol{int}}^{+} \boldsymbol{e}. \tag{26}$$

Next, we propose our second method for improving the system efficiency based on MPC.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

C. Vision-Based Model Predictive Control

Typically, in the MPC framework, the control signal is obtained by solving an optimization problem considering the future time instances of the process in each control step. The formulation of the problem includes a system model, a disturbance model, and the constraints of the task. Using the models, the future states are predicted, and the error between the future reference and state values are minimized also considering the additional constraints and weight parameters. We utilize generalized predictive control formulation in which the cost function aimed to be minimized is defined as

$$A(\boldsymbol{U}) = \sum_{j=1}^{N} \delta(j)[\hat{y}(t+j|t) - w(t+j)]^2 + \sum_{j=1}^{N} \gamma(j)[\Delta u(t+j-1)]^2. \quad (27)$$

Here, $\hat{y}(t+j|t)$ is the predicted system state at the future instance $t+j$ calculated at time t (considering both the system model and the effect of the disturbance), $w(t+j)$ is the value of the reference signal at time $t+j$, $\delta(j)$ and $\gamma(j)$ are the weighting parameters, N is the optimization window size, and

$$\Delta u(t) = u(t) - u(t-1) \quad (28)$$

$$\boldsymbol{U} = [u(t), u(t+1), \ldots u(t+j-1)] \quad (29)$$

where $u(t)$ is the system input. As traditionally used in visual servoing framework, $u(t)$ is applied as a velocity reference to the system. By minimizing the cost function in (27), the error between the reference and the state is aimed to be decreased to zero while minimizing the effect of disturbance and penalizing the total control.

The parameters $\delta(j)$ and $\gamma(j)$ determine the characteristic of the convergence. We chose to use an exponential weight for $\delta(j)$ as

$$\delta(j) = \alpha^{N-j} \quad (30)$$

where $0 < \alpha < 1$. By this way, the later values of the error are penalized more than the earlier values, and a smooth convergence can be achieved. We chose to use a constant positive $\gamma(j)$ value, which penalizes the actuator efforts equally throughout the optimization window.

The state prediction $\hat{y}(t+j|t)$ is calculated as a sum of the system model component y_s and the estimate of the disturbance on the state y_d as

$$\hat{y}(t+j|t) = y_s(t+j|t) + y_d(t+j|t). \quad (31)$$

We use a step response type system model, which is the sum of the effect of the efforts applied to the current state measurement

$$y_s(t+j|t) = y(t) + \sum_{i=1}^{j} g_i \Delta u(t+i). \quad (32)$$

Here, g_i are the model parameters and $y(t)$ is the last measured system state. In our implementation, we use a single g_i value (i.e., $g_1 = g_2 \ldots = g_j$) that we obtained experimentally by operating the system at the center of the workspace.

The effect of disturbance (y_d) can be incorporated in various ways. If disturbance is not directly measurable or predictable for the future values of the system state, then it can be assumed that its effect at the last control step will continue to be the same in the future steps. In this case, the disturbance can be estimated as the difference between the last measured state and the effect of control input on the previous state

$$y_d(t+j|t) = y(t) - (y(t-1) + g_1 \Delta u(t)). \quad (33)$$

Alternatively, if the reference signal is periodic, the disturbance is consistent throughout the periodic cycle, and the system already completed one full period, then the effect of the disturbance measured at the previous cycle can be used as an estimate

$$y_d(t+j|t) = y(t) - (y(t-\tau+j) + g_1 \Delta u(t-\tau+j)) \quad (34)$$

where τ is the number of control steps that correspond to one period of the reference signal. Such an approach is also used in robotic heart surgery where the disturbances are consistent and periodic [46].

With such a formulation of state prediction, every deviation from y_s will be considered as disturbance, which will be minimized by the optimization procedure. The optimization problem is

$$\arg\min_{\boldsymbol{U}} A(\boldsymbol{U}) \quad (35)$$

which has the following explicit solution:

$$\boldsymbol{U} = (\boldsymbol{P}^T \boldsymbol{P} + \boldsymbol{\gamma} \boldsymbol{I})^{-1} \boldsymbol{G}^T (\boldsymbol{W} - \boldsymbol{F}) \quad (36)$$

where I is an $N \times N$ identity matrix, and

$$\boldsymbol{P} = \begin{bmatrix} \alpha^{N-1} g_1 & 0 & 0 & & 0 \\ \alpha^{N-1} g_1 & \alpha^{N-2} g_2 & 0 & \cdots & 0 \\ \alpha^{N-1} g_1 & \alpha^{N-2} g_2 & \alpha^{N-3} g_3 & & 0 \\ & \vdots & & \ddots & \vdots \\ \alpha^{N-1} g_1 & \alpha^{N-2} g_2 & \alpha^{N-3} g_3 & \cdots & g_N \end{bmatrix} \quad (37)$$

$$\boldsymbol{F} = \begin{bmatrix} y_d(t+1|t) + y(t) \\ y_d(t+2|t) + y(t) \\ y_d(t+3|t) + y(t) \\ \vdots \\ y_d(t+N|t) + y(t) \end{bmatrix} \quad (38)$$

$$\boldsymbol{W} = \begin{bmatrix} w(t+1) \\ w(t+2) \\ w(t+3) \\ \vdots \\ w(t+N) \end{bmatrix}. \quad (39)$$

Even though control input is calculated for the next N control steps, only the first control input $u(t)$ is sent to the system, and the optimization problem is solved again with newly acquired data in the next step

$$\boldsymbol{V_o^{Cam}} = u(t) \quad (40)$$

$\boldsymbol{V_o^{Cam}}$ is projected to the actuator space with (18) and then (21).

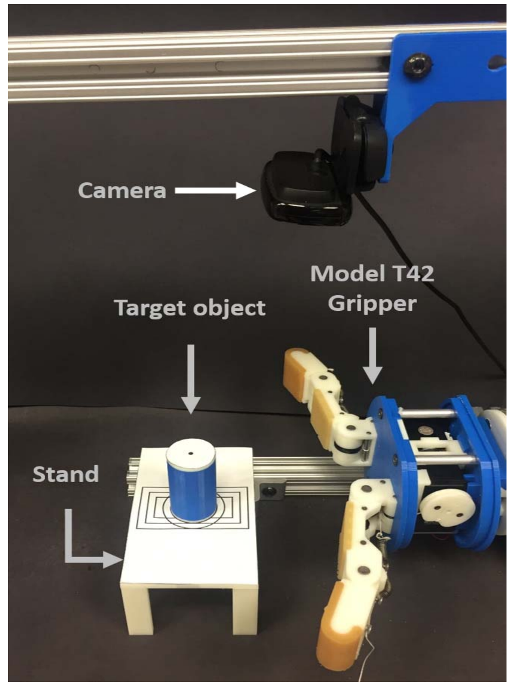
The advantages of using this MPC formulation over conventional image-based visual servoing (IBVS) rule are multifold. First, MPC takes into account the effect of disturbances on the system: any deviation from the step response model will be considered as a disturbance and will be minimized. By this, we do not only overcome the friction effects, but also the imperfections of the kinematics model. Second, we have much more control over the transient response of the system than we had with IBVS due to the additional parameters δ and γ. Therefore, smooth transitions can be achieved and torque limits can be imposed. Moreover, if the future values of the reference signal are known, convergence can be maintained faster since these values are taken into account in the optimization. Finally, with the MPC framework, additional constraints can be added to this optimization problem such as workspace constraints and task specific constraints, which could be desired for many precision manipulation scenarios. However, an explicit solution with additional constraints may not always exist and iterative solvers may be needed. In this case, high computational power is necessary as this optimization is run in every control step. Alternately, a self-triggering mechanism can be designed as in [44].

V. Experimental Results

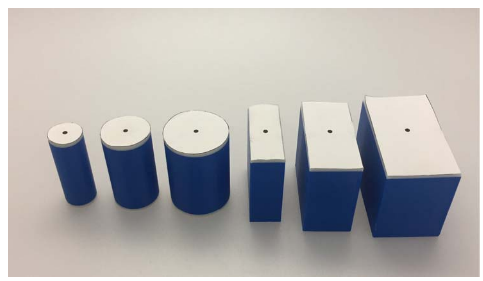
We conducted vision-based Cartesian positioning experiments with the Model T42 gripper using the Jacobian obtained in (7). The performances of the IBVS algorithm, correction matrix method, and MPC algorithm are tested and compared with the test setup presented in Fig. 6(a) using cylindrical and rectangular objects in various sizes as can be seen from Fig. 6(b). The camera is set to 512×1024 pixels resolution and 30 f/s capture rate. The distance of the camera to the object top surface is 18.5 cm. At the center of the workspace and at the object level, one pixel approximately corresponds to 0.23 mm. In each experiment, the object is placed on a stand, it is pinch grasped by the gripper, and then the stand is removed so that the object is manipulated without the plane support. The positions of the target objects are kept consistent in all experiments with a template on the stand, and the initial position of the stand was aligned with a static fixture. We evaluated the performance of the algorithms using step references and continuous periodic references.

A. Step Response Results

A sequence of step references is applied to the system for assessing the performance of our controller and the results are given in Fig. 7. For the cylindrical objects, the object starts at set point 1 presented in Fig. 8, and set points 2–10 are applied sequentially. When the error remains two pixels or less for 1 s, the next set point is sent to the system. For the rectangular objects, the same procedure is executed except that the objects start at set point 5, and points 6–8 are applied (set points 1–4 are not realizable for all rectangular objects since, while moving from the initial grasp configuration to these set points, the object slides beyond the operable workspace that a stable grasp cannot be maintained). For each object, IBVS, correction matrix, and MPC methods are applied. The optimization window size of MPC is set to 50 cycles.



(a)



(b)

Fig. 6. Experimental setup. (a) Layout. (b) Objects used in the experiments: cylinders with 2-, 3-, and 4-cm diameters, and rectangular prisms with dimensions 2×4, 3×5, and 4×6 cm. The objects' weights vary between 12 and 75 g.

For each method, the experiment is repeated five times, and the total time for visiting all the set points and the total travel distance in image space are presented for all the objects in Fig. 7. Also, an example of trajectories with the medium cylinder is presented for all the algorithms in Fig. 9. In Fig. 7, the MPC method is almost always more efficient than the conventional IBVS and the correction matrix method in terms of both the travel distance and time spent to complete the trajectory. The trajectories in Fig. 9 also show that our methods make the system follow a trajectory closer to the optimal one while traveling between the set points. This is due to the disturbance rejection mechanisms.

B. Effect of High Model Inaccuracy

The advantage of MPC can also be seen clearly at the challenging parts of the workspace. An example is presented in Fig. 1. In Fig. 1(a), the object is manipulated at the center of the workspace, where the derived simple models have small errors. In this case, the gains of both IBVS and vision-based MPC are tuned for the fastest convergence

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

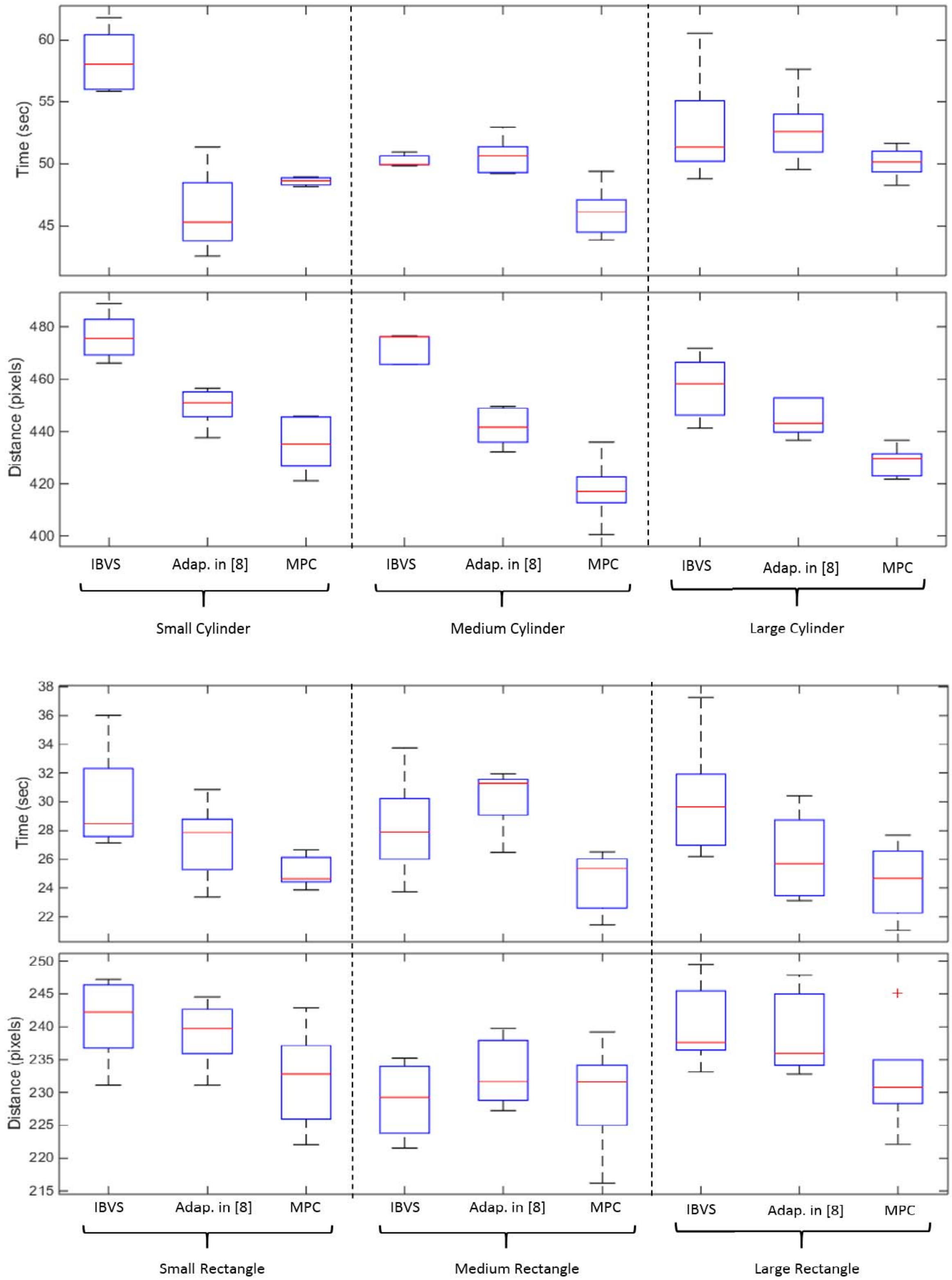


Fig. 7. Experiments with the cylindrical (top) and rectangular (bottom) objects. Time elapsed to complete the task and total distances traveled are presented. Each box represents five experiments with the same object.

with minimum deviation from the optimal trajectory in image space and minimum overshoot. The change of position of the object over time in x-direction is presented in Fig. 10. It can be seen that both algorithms make the object follow a close-to-optimum trajectory, whereas MPC provides faster convergence. In the case of Fig. 1(b), the experiment is conducted in a challenging part of the workspace, where the object roles over the tip of the finger, and the accuracy of the simple models drop significantly. In this experiment, the controller gains are kept the same with the previous trial.

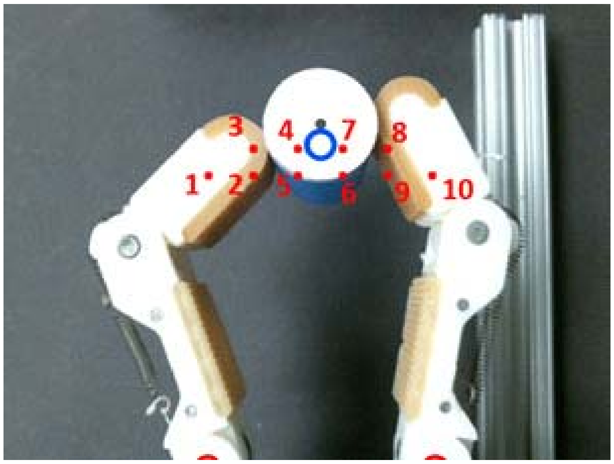


Fig. 8. Step references (red) and the circular reference (blue) used in the experiments.

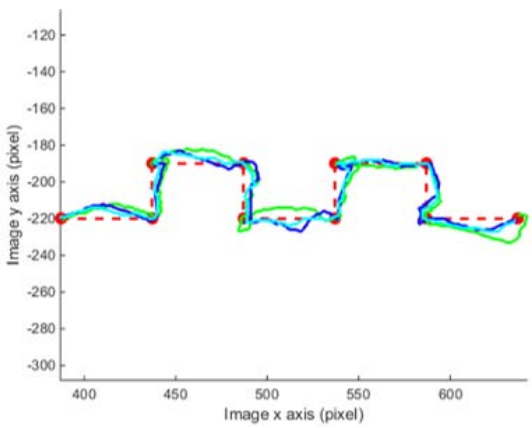


Fig. 9. Trajectories followed by the medium size cylinder with IBVS (green line), the correction matrix method (blue line), and vision-based MPC algorithm (cyan line). Red points signify the set points while dashed red lines are the optimal path in the image space. The set points are supplied sequentially.

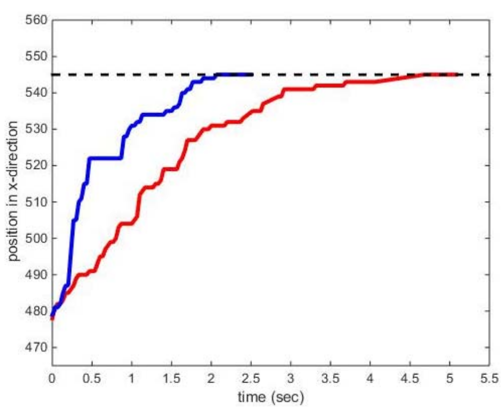


Fig. 10. Position of the tracked object feature in the ***x*-** direction for the experiment shown in Fig. 1(a) (the motion in the *y*- direction is negligible). Black dashed line: reference signal, red line: IBVS algorithm, and blue line: vision-based MPC algorithm.

The position of the object with respect to time is presented in Fig. 11 in the *x*- and *y*-directions. In this particularly difficult case, even though a simple combination of our models with

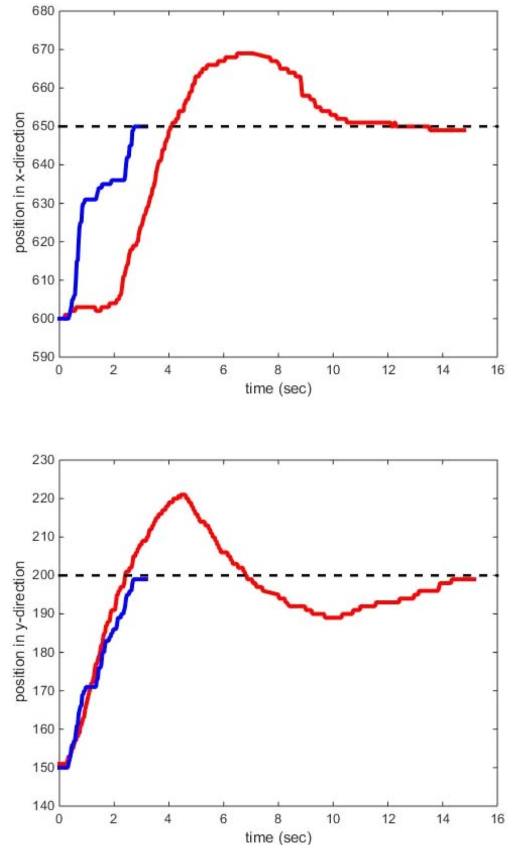


Fig. 11. Position of the tracked object feature in the *x*- and *y*- directions for the experiment shown in Fig. 1(b). Black dashed line: reference signal, red line: IBVS algorithm, and blue line: vision-based MPC algorithm.

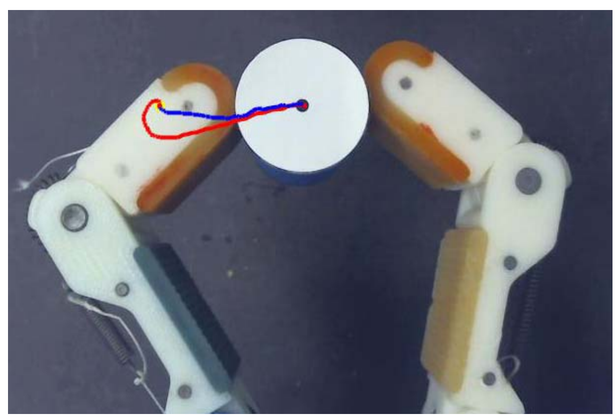


Fig. 12. Trajectory of the object for the part of the workspace where the modeling inaccuracies are high. Red line: IBVS algorithm and blue line: vision-based MPC algorithm. The trajectories are drawn on the image taken from the initial position of the system.

the traditional image-based visual servoing technique makes the system converge to the reference position in the steady state, the manipulation is much more efficient when the MPC-based approach is employed; MPC provides approximately five times faster convergence and makes the object follow a trajectory that is much closer to the optimum comparing to the IBVS algorithm. MPC also improves the robustness of the system since it prevents large trajectory deviations, and

TABLE I
EXPERIMENTAL RESULTS WITH THE CIRCULAR REFERENCE

Target object	IBVS	Corr. Matrix	MPC	MPC periodic
Small Cylinder	2.6	2.7 (-4%)	2.5 (4%)	2.3 (13%)
Medium Cylinder	2.7	2.8 (-4%)	2.4 (13%)	2.2 (22%)
Large Cylinder	2.7	3.1 (-12%)	2.4 (12%)	2.4 (12%)
Small Rectangle	2.8	2.9 (-3%)	2.7 (4%)	1.8 (56%)
Medium Rectangle	3.1	3.1 (0%)	2.7 (15%)	2.6 (19%)
Large Rectangle	2.9	3.0 (-3%)	2.9 (0%)	2.8 (%4)

Average errors are presented in pixels and improvements with respect to the traditional IBVS method are given in parenthesis in terms of percentages.

therefore, reduces the risk of dropping the object by keeping it in the manipulation workspace.

Another example can be seen from Fig. 12. Here, the object is moved toward a region where the modeling inaccuracies are high and the difference between the commanded velocity and actual velocity differs significantly as also demonstrated in Fig. 4. When the conventional visual servoing method is used, the system still converges, but deviates from the optimal trajectory by following an arc like path (as in Fig. 4). This results in a spiral motion, causes a significant inefficiency and brings a risk of moving out of the operable workspace. MPC compensates for the modeling inefficiency, and pushes the system toward the optimal trajectory.

C. Periodic Signal Tracking

The circular reference shown in Figs. 7 with 15 pixel radius and period of 20 s is applied to the system for three periods. For this set of experiments, MPC with two different disturbance estimation methods are evaluated. In the first case, the effect of the disturbance is calculated by the information of the previous control step as in (33). In the second case, the effect of disturbance is calculated by the information of the previous period as in (34). The performance of the algorithms is evaluated with all the target objects, and the average error values over the three periods are presented in Table I. Here, it is observed that the correction matrix method degrades the performance. The reason is that this method necessitates larger amount of motions for estimating the velocity direction and for correcting it; small motions in the image space results with quantization errors, and correction cannot be conducted accurately. MPC methods, however, improve the performance of the system significantly. We observe that the periodic MPC enhances the tracking performance even more as it has a more accurate estimation of the disturbance for the future steps of the optimization.

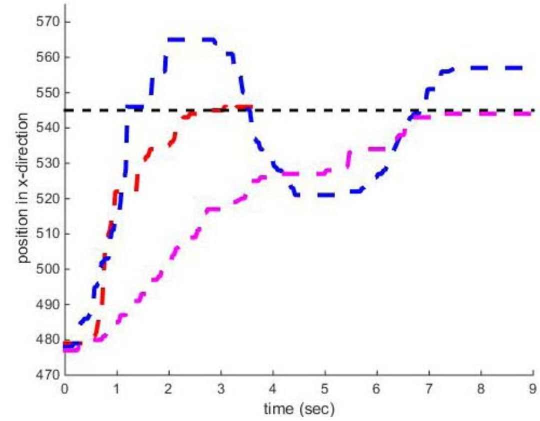


Fig. 13. Position of the tracked object in the x- direction for varying α values. Red dashed line: $\alpha = 0.668$, blue dashed line: $\alpha = 0.670$, and magenta dashed line: $\alpha = 0.666$. The γ value is set to 0.002 in these experiments.

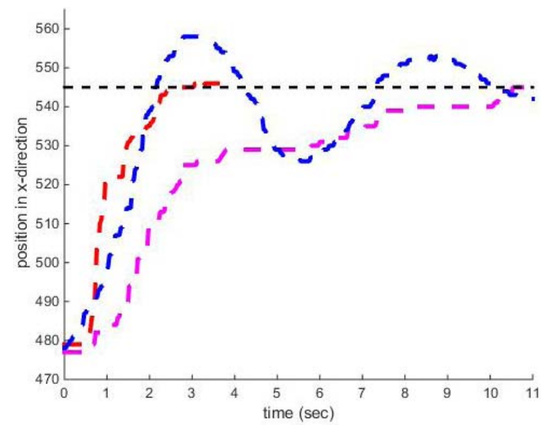


Fig. 14. Position of the tracked object in the x- direction for varying γ values. Red dashed line: $\gamma = 0.002$, blue dashed line: $\gamma = 0.0017$, and magenta dashed line: $\gamma = 0.003$. The α value is set to 0.668 in these experiments.

D. Effect of MPC Parameters

The α parameter in (30) and γ and N parameters in (27) are used to adjust the behavior of the MPC algorithm. In Figs. 13 and 14, the convergence of the system is presented for varying α and γ values, respectively, for the trajectory presented in Fig. 1(a). It can be seen from Fig. 13 that, while keeping γ values the same, high α values effectively cause an under damped system resulting in faster convergence together with overshoot and oscillations. For lower values of α, the later values of the error are penalized more than the earlier ones, therefore the system has a smoother, but slower convergence. In Fig. 14, we can see that high γ values slow down the system response as it penalizes the sum of inputs in the overall optimization window. Low γ values do not necessarily result in a faster convergence to the desired set point, since using the same α value still penalizes the earlier and later values of the error in the same way. However, much higher settling time is observed. In our experience, γ parameter is very useful for the precision manipulation controller for keeping the inputs within the bandwidth of the system, and therefore maintaining the stability.

The window size parameter N cannot be tuned independent of the other parameters: N determines the number of samples considered in the optimization process. The values of both α and γ parameters depend on N as their effect is distributed among the samples.

VI. Conclusion

In this paper, we present a strategy for conducting efficient vision-based in-hand manipulation with simple process models. The performance degradations due to the modeling inaccuracies are minimized using a correction matrix method and the MPC framework. The experimental results show that correction matrix and MPC methods improve system efficiency by providing faster and smoother step response. MPC shows a superior performance and also enhances the performance in signal tracking, where correction matrix method fails. The use of periodic disturbance models in MPC provides even better results in periodic signal tracking, since the effect of disturbance in the future control steps can be estimated more accurately.

MPC framework is also ideal for introducing workspace and task constraints to the process. In our next work, we aim to develop vision-based learning algorithms for detecting the workspace constraints of the system and integrating them to the optimization procedure. We also plan to utilize our strategy for scenarios that require interactions with the surroundings, and analyze the performance of the system when external disturbances exist.

We also investigate conducting controlled sliding using vision feedback. Our methods with disturbance rejection are robust to object sliding (which occasionally occurs in our experiments) even though we do not explicitly consider it in our models or control scheme. However, controlling sliding means, in the Model T42 example, determining the controllable 2-D submanifold of the 3-D workspace; therefore, this ability can extend the workspace and the system dexterity substantially.

References

[1] R. R. Ma and A. M. Dollar, "On dexterity and dexterous manipulation," in *Proc. IEEE 15th Int. Conf. Adv. Robot. (ICAR)*, Jun. 2011, pp. 1–7.

[2] K. Hertkorn, M. A. Roa, and C. Borst, "Planning in-hand object manipulation with multifingered hands considering task constraints," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 617–624.

[3] R. Michalec and A. Micaelli, "Stiffness modeling for multi-fingered grasping with rolling contacts," in *Proc. 10th IEEE-RAS Int. Conf. Humanoid Robots (Humanoids)*, Dec. 2010, pp. 601–608.

[4] A. M. Okamura, N. Smaby, and M. R. Cutkosky, "An overview of dexterous manipulation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 1. Apr. 2000, pp. 255–262.

[5] A. Bicchi, C. Melchiorri, and D. Balluchi, "On the mobility and manipulability of general multiple limb robots," *IEEE Trans. Robot. Autom.*, vol. 11, no. 2, pp. 215–228, Apr. 1995.

[6] F. Chaumette and S. Hutchinson, "Visual servo control. I. Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, Dec. 2006.

[7] F. Chaumette and S. Hutchinson, "Visual servo control. II. Advanced approaches," *IEEE Robot. Autom. Mag.*, vol. 14, no. 1, pp. 109–118, Mar. 2007.

[8] B. Calli and A. M. Dollar, "Vision-based precision manipulation with underactuated hands: Simple and effective solutions for dexterity," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 1012–1018.

[9] R. Deimel and O. Brock, "A novel type of compliant and underactuated robotic hand for dexterous grasping," *Int. J. Robot. Res.*, vol. 35, nos. 1–3, pp. 161–185, 2016.

[10] L. U. Odhner *et al.*, "A compliant, underactuated hand for robust manipulation," *Int. J. Robot. Res.*, vol. 33, no. 5, pp. 736–752, Apr. 2014.

[11] L. U. Odhner, R. R. Ma, and A. M. Dollar, "Open-loop precision grasping with underactuated hands inspired by a human manipulation strategy," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 3, pp. 625–633, Jul. 2013.

[12] L. Birglen, T. Laliberté, and C. M. Gosselin, *Underactuated Robotic Hands*. Berlin, Germany: Springer-Verlag, 2008.

[13] R. Ma and A. Dollar, "Yale openhand project: Optimizing open-source hand designs for ease of fabrication and adoption," *IEEE Robot. Autom. Mag.*, vol. 24, no. 1, pp. 32–40, Mar. 2017.

[14] B.-H. Kim, B.-J. Yi, S.-R. Oh, and I. H. Suh, "Independent finger and independent joint-based compliance control of multifingered robot hands," *IEEE Trans. Robot. Autom.*, vol. 19, no. 2, pp. 185–199, Apr. 2003.

[15] D. Prattichizzo, M. Malvezzi, M. Aggravi, and T. Wimböck, "Object motion-decoupled internal force control for a compliant multifingered hand," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 1508–1513.

[16] B. C. Calli and A. M. Dollar, "Vision-based model predictive control for within-hand precision manipulation with underactuated grippers," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 2839–2845.

[17] M. T. Mason and J. K. J. Salisbury, *Robot Hands and The Mechanics of Manipulation*. Cambridge, MA, USA: MIT Press, 1985.

[18] L. U. Odhner and A. M. Dollar, "Dexterous manipulation with underactuated elastic hands," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2011, pp. 5254–5260.

[19] Y. Li and I. Kao, "A review of modeling of soft-contact fingers and stiffness control for dextrous manipulation in robotics," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 3. May 2001, pp. 3055–3060.

[20] N. Rojas, R. R. Ma, and A. M. Dollar, "The GR2 gripper: An underactuated hand for open-loop in-hand planar manipulation," *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 763–770, Jun. 2016.

[21] R. R. Ma, N. Rojas, and A. M. Dollar, "Spherical hands: Toward underactuated, in-hand manipulation invariant to object size and grasp location," *J. Mech. Robot.*, vol. 8, no. 6, p. 061021, 2016.

[22] M. Yashima, Y. Shiina, and H. Yamaguchi, "Randomized manipulation planning for a multi-fingered hand by switching contact modes," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2, Sep. 2003, pp. 2689–2694.

[23] A. Bicchi, M. Gabiccini, and M. Santello, "Modelling natural and artificial hands with synergies," *Philos. Trans. Roy. Soc. London B, Biol. Sci.*, vol. 366, no. 1581, pp. 3153–3161, 2011.

[24] L. U. Odhner, R. R. Ma, and A. M. Dollar, "Precision grasping and manipulation of small objects from flat surfaces using underactuated fingers," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2012, pp. 2830–2835.

[25] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem," in *Proc. Robot. Sci. Syst. Workshop-Sens. Adapt. Real World*, 2007, pp. 1–6.

[26] M. Ciocarlie, C. Goldfeder, and P. Allen, "Dimensionality reduction for hand-independent dexterous robotic grasping," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, Oct. 2007, pp. 3270–3275.

[27] M. G. Catalano, G. Grioli, E. Farnioli, A. Serio, C. Piazza, and A. Bicchi, "Adaptive synergies for the design and control of the pisa/IIT SoftHand," *Int. J. Robot. Res.*, vol. 33, no. 5, pp. 768–782, Apr. 2014.

[28] G. Gioioso, G. Salvietti, M. Malvezzi, and D. Prattichizzo, "An object-based approach to map human hand synergies onto robotic hands with dissimilar kinematics," in *Robotics: Science and Systems VIII*. Cambridge, MA, USA: MIT Press, 2013, pp. 97–104.

[29] O. Brock *et al.*, "Transferring synergies from neuroscience to robotics? Comment on 'Hand synergies: Integration of robotics and neuroscience for understanding the control of biological and artificial hands' by M. Santello et al," *Phys. Life Rev.*, vol. 1, pp. 27–32, Aug. 2016.

[30] F. Chaumette, S. Hutchinson, and P. Corke, "Visual servoing," in *Springer Handbook of Robotics*. Berlin, Germany: Springer-Verlag, 2016.

[31] D. Kragic and H. I. Christensen, "Survey on visual servoing for manipulation," Centre Auto. Syst., Royal Inst. Technol. Stockholm, Tech. Rep. ISRNKTH/NA/P–02/01–SE, 2002.

[32] K. Wang, Y. Liu, and L. Li, "Visual servoing trajectory tracking of nonholonomic mobile robots without direct position measurement," *IEEE Trans. Robot.*, vol. 30, no. 4, pp. 1026–1035, Aug. 2014.
[33] V. Guelpa, G. J. Laurent, B. Tamadazte, P. Sandoz, N. Le Fort-Piat, and C. Clévy, "Single frequency-based visual servoing for microrobotics applications," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 4274–4279.
[34] C. A. Jara, J. Pomares, F. A. Candelas, and F. Torres, "Optimal control for robot-hand manipulation of an object using dynamic visual servoing," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2014, pp. 89–94.
[35] Q. Li, C. Elbrechter, R. Haschke, and H. Ritter, "Integrating vision, haptics and proprioception into a feedback controller for in-hand manipulation of unknown objects," in *Proc. IEEE Int. Conf. Intell. Robot. Syst.*, Nov. 2016, pp. 2466–2471.
[36] M. Jagersand, O. Fuentes, and R. Nelson, "Experimental evaluation of uncalibrated visual servoing for precision manipulation," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, vol. 4, Apr. 1997, pp. 2874–2880.
[37] M. Jägersand, "Visual servoing using trust region methods and estimation of the full coupled visual-motor Jacobian," in *Proc. IASTED Appl. Robot. Control*, 1996, pp. 105–108.
[38] E. F. Camacho and C. A. Bordons, *Model Predictive Control*. London, U.K.: Springer-Verlag, 2007.
[39] Z. Li, C. Yang, C. Su, J. Deng, and W. Zhang, "Vision-based model predictive control for steering of a nonholonomic mobile robot," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 2, pp. 553–564, Mar. 2016.
[40] C. J. Ostafew, A. P. Schoellig, T. D. Barfoot, and J. Collier, "Learning-based nonlinear model predictive control to improve vision-based mobile robot path tracking," *J. Field Robot.*, vol. 33, no. 1, pp. 133–152, 2015.
[41] G. Franz and W. Lucia, "A model predictive control scheme for mobile robotic vehicles in dynamic environments," in *Proc. IEEE Conf. Decision Control*, Dec. 2013, pp. 5752–5757.
[42] J. C. de la Cárdenas, A. S. García, S. S. Martínez, J. G. García, and J. G. Ortega, "Model predictive position/force control of an anthropomorphic robotic arm," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Mar. 2015, pp. 326–331.
[43] M. J. Powell, E. A. Cousineau, and A. D. Ames, "Model predictive control of underactuated bipedal robotic walking," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 5121–5126.
[44] S. Heshmati-Alamdari, A. Eqtami, G. C. Karras, D. V. Dimarogonas, and K. J. Kyriakopoulos, "A self-triggered visual servoing model predictive control scheme for under-actuated underwater robotic vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 3826–3831.
[45] R. Ginhoux, J. A. Gangloff, M. F. de Mathelin, L. Soler, M. M. A. Sanchez, and J. Marescaux, "Beating heart tracking in robotic surgery using 500 Hz visual servoing, model predictive control and an adaptive observer," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 1. Apr. 2004, pp. 274–279.
[46] R. Ginhoux, J. Gangloff, M. F. de Mathelin, L. Soler, M. M. A. Sanchez, and J. Marescaux, "Active filtering of physiological motion in robotized surgery using predictive control," *IEEE Trans. Robot.*, vol. 21, no. 1, pp. 67–79, Feb. 2005.
[47] C. M. Best, M. T. Gillespie, P. Hyatt, L. Rupert, V. Sherrod, and M. D. Killpack, "A new soft robot control method: Using model predictive control for a pneumatically actuated humanoid," *IEEE Robot. Autom. Mag.*, vol. 23, no. 3, pp. 75–84, Sep. 2016.
[48] L. U. Odhner and A. M. Dollar, "Stable, open-loop precision manipulation with underactuated hands," *Int. J. Robot. Res.*, vol. 34, no. 11, pp. 1347–1360, 2015.
[49] D. Prattichizzo and J. C. Trinkle, "Grasping," in *Springer Handbook of Robotics*. Berlin, Germany: Springer-Verlag, 2008, pp. 671–700.

Berk Calli (M'16) received the Bachelor of Science and Master of Science degrees in the mechatronic program from Sabanci University, Istanbul, Turkey, and the Ph.D. degree from the Delft University of Technology, Delft, The Netherlands, in 2015. His master thesis was on integrated visual servoing and force control for robotic manipulation. His Ph.D. dissertation focuses on active sensing solutions for improving the performance of the grasp synthesis algorithms.

He is currently with GRAB Laboratory, Yale University, New Haven, CT, USA, working on vision-based manipulation, dexterous manipulation, and manipulation benchmarking.

Aaron M. Dollar (SM'13) received the B.S. degree in mechanical engineering from the University of Massachusetts at Amherst, Amherst, MA, USA, and the M.S. and Ph.D. degrees in engineering sciences from Harvard University, Cambridge, MA, USA.

He conducted two years of postdoctoral research at the MIT Media Laboratory, Cambridge, MA, USA. He is currently the John J. Lee Associate Professor of Mechanical Engineering and Materials Science with Yale University, New Haven, CT, USA. His research topics include human and robotic grasping and dexterous manipulation, mechanisms and machine design, and assistive and rehabilitation devices including upper-limb prosthetics and lower-limb orthoses.

Dr. Dollar was a recipient of the 2013 DARPA Young Faculty Award, the 2011 AFOSR Young Investigator Award, the 2010 Technology Review TR35 Young Innovator Award, and the 2010 U.S. National Science Foundation CAREER Award.