Randomized Kinodynamic Planning for Robust Visual Servoing

Moslem Kazemi, Kamal K. Gupta, *Member, IEEE*, and Mehran Mehrandezh, *Member, IEEE*

***Abstract*—We incorporate a randomized *kinodynamic* path planning approach with image-based control of a robotic arm equipped with an in-hand camera. The proposed approach yields continuously differentiable camera trajectories by taking camera dynamics into account, while accounting for a critical set of image and physical constraints at the planning stage. The proposed planner explores the camera *state* space for permissible trajectories by iteratively extending a search tree in this space and simultaneously tracking these trajectories in the robot configuration space. The planned camera trajectories are projected into the image space to obtain desired feature trajectories which are then tracked using an image-based visual servoing scheme. We validate the effectiveness of the proposed framework in incorporating the aforementioned constraints through a number of visual servoing experiments on a six-degree-of-freedom robotic arm. We also provide empirical results that demonstrate its performance in the presence of uncertainties, and accordingly suggest additional planning strategies to increase robustness with respect to possible deviations from planned trajectories.**

***Index Terms*—Kinodynamic path planning, robotic manipulators, sampling-based path planning, visual servoing.**

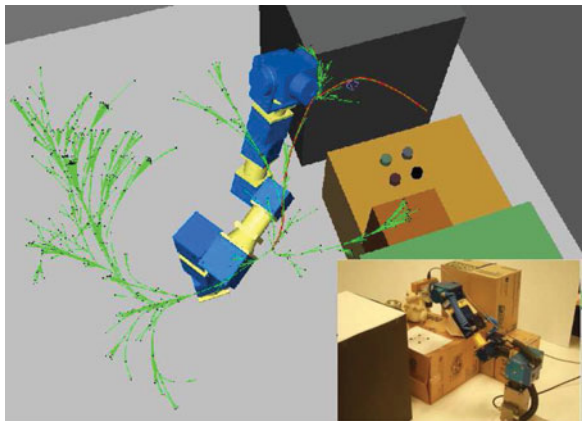


Fig. 1. Our proposed randomized kinodynamic planner explores the camera state space for permissible trajectories by iteratively extending a search tree in this space and simultaneously tracking these trajectories in the robot configuration space.

I. Introduction

We address the problem of path planning for image-based control of robotic arms (see Fig. 1) with the aim of extending the robustness of classical image-based visual servoing (IBVS) techniques [1] to image and physical constraints. In contrast with position-based visual servoing (PBVS), where the control is performed in the task space based on the 3-D information retrieved from image, in IBVS techniques, the feedback is defined based on image features, and the control loop is closed directly within the image. This results in a more robust control in the presence of calibration and modeling errors [2], and hence adds to the popularity of IBVS methods.

In [3], through simple yet effective examples, Chaumette outlined the potential problems of stability and convergence of IBVS techniques: singularities in image Jacobian leading to an unstable behavior, and reaching local minima due to the existence of multiple camera poses yielding the same terminal image of the target. Moreover, in IBVS techniques, there is no direct control over the image/camera/robot trajectories induced by the servoing loop in the image and physical spaces. Therefore, these trajectories might violate the image and/or physical constraints. The image constraints normally encountered in practice include 1) field of view limits, i.e., the target may become invisible because it is outside of the camera's field of view; 2) occlusion constraint, i.e., the target may be occluded due to obstacles, robot body, or self-occlusion, and physical constraints include robot kinematics, such as joint limits, singularities in robot Jacobian, robot/camera dynamic constraints, and collision with obstacles or self-collision. These constraints are discussed with more details in [4].

Since Chaumette's article [3] on the stability and convergence problems of IBVS techniques, research efforts in visual servoing have been devoted to incorporating the above image and physical constraints into the reactive visual servoing loop. Earlier works include *partitioned* and *switched* strategies. In contrast with partitioned strategies (e.g., [5] and [6]), where certain degrees of freedom (DOFs) are controlled via IBVS, while others are controlled via PBVS, switched strategies consist of a set of visual servo controllers along with a switching rule (e.g., [7] and [8]). Partitioned techniques take advantage of both IBVS and PBVS techniques in avoiding some of the aforementioned constraints, while switched strategies enlarge the stability region of classical visual servoing techniques by switching between a set of unstable controllers to make the overall system stable.

Manuscript received June 19, 2012; revised December 6, 2012; accepted May 16, 2013. This paper was recommended for publication by Associate Editor E. Marchand and Editor G. Oriolo upon evaluation of the reviewers' comments. The work of K. Gupta and M. Mehrandezh was supported by Natural Sciences and Engineering Research Council Discovery grants.

M. Kazemi is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: moslemk@andrew.cmu.edu).

K. Gupta is with the School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: kamal@sfu.ca).

M. Mehrandezh is with the Faculty of Engineering and Applied Science, University of Regina, Regina, SK S4S 0A2, Canada (e-mail: mehran.mehrandezh@uregina.ca).

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/TRO.2013.2264865

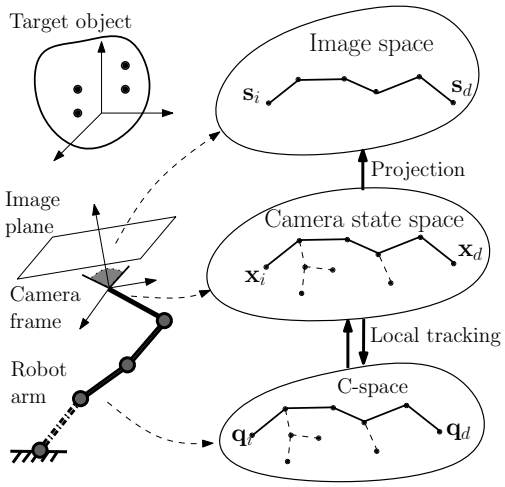


Fig. 2. Our proposed planner alternatively explores the camera state space and robot configuration space for feasible camera/robot paths to obtain feasible feature trajectories in the image space.

Pertinent literature on partitioned and switched strategies indicates that only a small subset of the aforementioned constraints can be incorporated (see the review in [4]). Incorporating a global and general path planning strategy into the visual servo loop seems a promising effort toward addressing all the aforementioned constraints, especially in complex visual servoing scenarios. The main idea behind path planning for visual servoing [4], [9] is to plan and generate *feasible* image feature trajectories while accounting for the constraints, and then to servo the robot along the planned trajectories. Overall, this results in a more robust servoing process with respect to violation of image and physical constraints.

In this paper, we incorporate a global randomized kinodynamic path planning approach with image-based control for a robotic arm equipped with an in-hand camera. The proposed approach yields planned C^1-smooth camera trajectories by taking the camera dynamics into account as well as accounting for a critical set of image and physical constraints: 1) maintaining continuous visibility of the target; 2) avoiding visual occlusion of target features caused by the workspace obstacles, robot's body, or the target itself; 3) avoiding robot collision with physical obstacles or self-collision; and 4) joint limits. The proposed planner explores the camera *state space* (i.e., a 12-D vector space consisting of camera poses and velocity screws) for permissible trajectories by iteratively extending a search tree in this space and simultaneously tracking these trajectories in the robot configuration space (see Fig. 2). The planned camera trajectories could then be executed in different ways. An ideal and challenging way would be to design a controller that reconciles errors in both joint space and image space. This is a future research topic. More simply, one could project the camera trajectory on to the image space to obtain the desired image feature trajectories and then adopt an IBVS controller (controller I) to track the feature trajectories at the execution stage. Alternatively, since we do have the joint space trajectories, these could be directly executed using a joint space controller (controller II). Intuitively, we expect controller I would yield smaller errors in image space at the expense of larger errors in joint space, and controller II would yield smaller errors in joint space at the expense of larger errors in the image space. One could also employ a combination of the two controllers, using controller II for the initial portion of the planned trajectory followed by controller I for the last "segment" of the trajectory to eliminate image space errors at the desired goal feature locations position. We illustrate our approach with controller I but have included some empirical comparisons between the two controllers vis-a-vis image space, and joint space errors in the presence of calibration and model uncertainties.

Our proposed framework is general and highly flexible through which all the image and physical constraints can be easily incorporated at the planning stage, which is a contribution in its own right over existing techniques none of which considers all these constraints. The effectiveness of the proposed approach has been validated through real experiments on a robotic arm with an eye-in-hand configuration that performs servoing tasks in complex environments. A short version of this work has been presented in [10].

In the proposed framework, we assume that obstacles in the workspace are known. We also assume that the 3-D model of the target object and camera intrinsic parameters are known *a priori*. In practice, however, accurate calibration and modeling may be difficult. Although this study does not account for calibration and modeling errors in the planning stage, we provide empirical results that demonstrate its performance in the presence of uncertainties. We also suggest additional planning strategies to increase robustness with respect to possible deviations from planned trajectories during execution. Incorporating these schemes into the current framework is the subject of our future research.

The rest of this paper is organized as follows. In Section II, we discuss related work and contrast the capabilities of our framework with those of the previous works. In Section III, we give an overview of the problem, our assumptions, and the solution methodology. In Section IV, we discuss our proposed planning strategy. The software architecture and implementation of the proposed approach is explained in Section V. Section VI presents and discusses the experimental results performed on a 6-DOF robotic arm. In Section VII, we provide empirical results comparing the effect of various uncertainties/modeling errors on image trajectory tracking (as employed to execute the planned trajectories in this study) in contrast with a simple joint trajectory tracking strategy.

II. Related Work

Literature on path planning for visual servoing is vast. Here, we limit ourselves to major works in this area, and encourage the reader to study the comprehensive survey in [4].

Avoiding field of view limits and robustness to camera calibration and modeling errors motivated a number of techniques aimed to interpolate a path *directly* in the image space between the initial and desired images without using any knowledge of camera calibration or target model. Various results from projective geometry have been applied in this context including

epipolar geometry [11], [12], projective homography [13]–[17], and projective invariance [2]. The main advantage of these approaches is their insensitivity to camera calibration and/or object model errors. One of the difficulties in these techniques is that the planned path in the image may not correspond to a feasible camera motion. Moreover, since the planning is done directly in the image space, satisfying physical constraints (e.g., joint limits and obstacles) will be very challenging through such approaches, lending themselves to be ineffective in complex visual servoing scenarios.

Potential fields (as in [18]) have been employed in the context of visual servoing in the face of constraints (e.g., field of view and joint limits [9], or obstacle avoidance behavior [7]) by applying repulsive potentials in the image, joint, and/or task spaces. One of the main advantages of potential field-based approaches is that they can be employed in real-time applications. As an inherent deficiency of potential field-based path planning methods, the aforementioned strategies are prone to getting trapped in local minima. As a remedy, global navigation functions [19] could be employed instead. For example, a global stabilizing strategy using navigation functions is presented in [20] that guarantees convergence to a visible goal from almost every initial visible configuration, while maintaining viability of all features along the way. However, one should note that constructing such navigation functions is limited to very simple scenarios only.

There is also a substantial body of literature aimed at finding globally optimal paths with respect to various costs (e.g., distance from the image boundary, length of the path traversed by the robot, energy expenditure, etc.). These include planning closed-form collineation paths corresponding to minimum energy and minimum acceleration camera paths [13], optimization over polynomial parametrization of the scaled camera paths [21], [22], convex optimization [23], [24], or techniques based on optimal control theory such as visual motion planning using Lagrange multipliers [25], and geodesic techniques [26]–[29]. Although the aforementioned techniques provide better insight into the complexity of the problem in finding optimal paths, they are more or less limited to simple scenarios. Introducing global image/physical constraints greatly adds to the complexity of the optimization problem, and hence, accounting for such constraints through the aforementioned frameworks is either difficult or highly detrimental to their time complexity.

The convergence problems of potential field-based techniques on one hand and the expensive cost of the aforementioned optimization-based techniques on the other hand motivate the need for general, yet global path planning approaches such as randomized sampling-based techniques [19], [30]. For example, in [31], a probabilistic roadmap approach has been utilized to plan minimal-occlusion paths for a camera with respect to a target object, which requires explicit computation of the boundary of visible and occluded regions. They also employed a dynamic collision checking strategy to check for the field of view limits along the edges of the roadmap.

Inspired by the work in [32] on path planning with general end-effector constraints, we devised a tree-based randomized path planning approach [33] to incorporate global image/physical constraints for robust execution of servoing tasks with respect to violation of such constraints. The path planning used in [33], however, was purely kinematic and results in discrete feature trajectories that require being properly time scaled so they can then be tracked at the execution stage. More specifically, we used cubic splines to interpolate and time scale the discrete feature trajectories [33].

Such time scaling can also be achieved either by synthesizing a controller with the desired dynamic characteristics of the underlying system to execute the planned trajectories (e.g., [34]) or by performing optimization over the kinematically planned paths to satisfy the dynamic constraints (e.g., [35] and [36]). However, notwithstanding the time-scaling method used, the robot may not be able to execute the trajectories produced by the kinematic planner due to its limits on actuator forces/torques and/or dynamic effects of machine vision system (see [37]) which are not taken into account at the planning phase.

Our work addresses this important issue at the planning stage by performing kinodynamic planning for the camera in its state space, while taking its dynamic constraints into account. Unlike other pure kinematic path planning approaches for visual servoing, which usually require an additional (cubic) spline interpolation step to properly time scale the feature trajectories and obtain smooth trajectories (see, e.g., [9]), the solution trajectories that are obtained using our proposed kinodynamic planning framework are (by construction) C^1-smooth and require no further postprocessing to be executed smoothly using an IBVS technique preventing a stop-and-go motion, an effect which is inevitable in kinematic planning as observed in our previous work [33].

III. Preliminaries and Overview

A. Notations

Considering Fig. 3, let $\mathcal{F}_o$ be the frame attached to the target, and $\mathcal{F}_k$ denote the camera frame at a pose along a camera trajectory. Let $\mathcal{P}_j$ for $j = 1, \ldots, n$, be a 3-D point on the target object with homogeneous coordinates $[X_j \; Y_j \; Z_j 1]^T$ in the camera frame $\mathcal{F}_k$. Given a frontal pin-hole perspective camera model, the projection of $\mathcal{P}_j$ into the image plane of $\mathcal{F}_k$ is given as a point with homogeneous normalized coordinates $m_j = [x_j \;\; y_j 1]^T$ where

$$x_j = \frac{1}{Z_j} X_j \quad \text{and} \quad y_j = \frac{1}{Z_j} Y_j. \tag{1}$$

The corresponding pixel coordinates can be calculated as $p_j = [u_j, \; v_j, \; 1]^T = A m_j$, where matrix A contains the intrinsic parameters of camera as

$$A = \begin{bmatrix} f p_u & 0 & u_0 \\ 0 & f p_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

In (2), f denotes the focal length, u_0 and v_0 are the pixel coordinates of camera principal point, and p_u and p_v are the number of pixels per unit distance in image coordinates.

We use the 2-D images of a number of 3-D points $\mathcal{P}_j$ for $j = 1, \ldots, n$ as image features to represent the solution trajectories

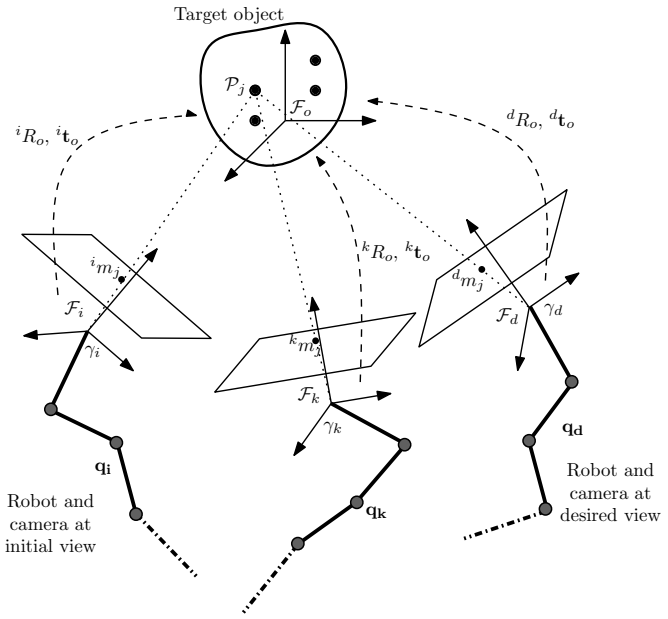


Fig. 3. Robot/camera at the initial, intermediate, and desired configurations w.r.t. the target object.

in the image. Hence, a point $\mathbf{s}$ along a feature trajectory is then represented as a vector $\mathbf{s} = [u_1\ v_1\ \ldots\ u_n\ v_n]^T$, where u_j and v_j are the image pixel coordinates of 3-D points $\mathcal{P}_j$ for $j = 1, \ldots, n$.

Given the coordinates of four or more point features in the object frame $\mathcal{F}_o$ and knowing the correspondences between their initial and desired image features, using numerical iterative techniques such as that in [38] and [39], one could compute the transformation between the initial and desired camera frames, $\mathcal{F}_i$ and $\mathcal{F}_d$, respectively, and the target object frame $\mathcal{F}_o$, i.e., ${}^iR_o$, ${}^it_o$, ${}^dR_o$, and ${}^dt_o$ (see Fig. 3).

B. Problem Formulation and Solution Methodology

The aim of path planning for visual servoing is to plan feature trajectories $\mathbf{s}(t)$ for $t \in [0, t_f]$ in the image space between initial and desired image features, $\mathbf{s}(0) = \mathbf{s}_i$ and $\mathbf{s}(t_f) = \mathbf{s}_d$, extracted from the images taken at the initial and final desired camera poses, respectively. The final time t_f is usually determined based on a time-scaling strategy imposed by the underlying planning algorithm.

Our proposed planning approach is summarized in the following two steps.

Step 1 (Planning feasible robot/camera trajectories): first, we plan a feasible camera trajectory $\Gamma(t)$ for $t \in [0, t_f]$, in the camera state space between the initial and desired camera states, $\Gamma(0) = \mathbf{x}_i$ and $\Gamma(t_f) = \mathbf{x}_d$, respectively. The state vector $\mathbf{x}(t)$ of the camera is defined as

$$\mathbf{x}(t) = [\,\mathbf{p}(t)^T\,\mathbf{h}(t)^T\,\mathbf{v}(t)^T\,\boldsymbol{\omega}(t)^T\,]^T \tag{3}$$

where $\mathbf{p} = [p_x\ p_y\ p_z]^T$ is the position of the camera frame, and $\mathbf{h}(t)$ denotes the quaternion representation of the camera orientation w.r.t. the object frame $\mathcal{F}_o$ (which is assumed to coincide with world coordinate frame without losing generality). $\mathbf{v} = [v_x\ v_y\ v_z]^T$ and $\boldsymbol{\omega} = [\omega_x\ \omega_y\ \omega_z]^T$ denote the camera linear and angular velocities, respectively. This makes the state $\mathbf{x}(t)$ a 13-D vector, but the state space is 12-D because of the constraint that the quaternion must be of unit norm.

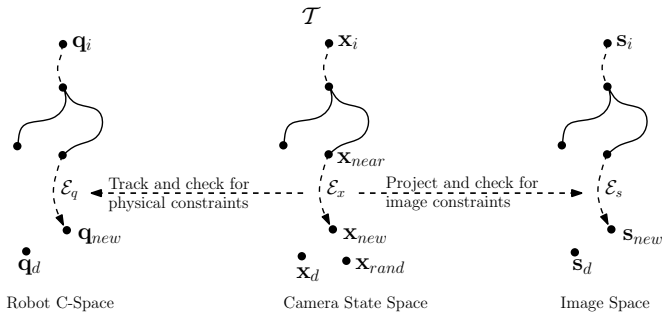


Fig. 4. Alternate camera state space and joint space exploration.

The camera trajectory is obtained by alternate exploration of the camera state space and robot joint space, which corresponds to a trackable robot trajectory $\mathbf{q}(t)$ for $t \in [0, t_f]$ in the joint space between the start and goal configurations, $\mathbf{q}(0) = \mathbf{q}_i$ and $\mathbf{q}(t_f) = \mathbf{q}_d$, respectively. The proposed planner explores the camera state space for permissible trajectories by iteratively extending a search tree in the camera state space and simultaneously tracking these trajectories in the robot joint space. The proposed planning approach is explained in more detail in Section IV.

Step 2 (Computing image feature trajectories): the camera trajectory $\Gamma(t)$ planned in Step 1 is then utilized to project the 3-D target feature points into the image space and obtain the feature trajectories $\mathbf{s}(t)$ between the initial and desired image features in the image space.

For a given camera frame, denoted by a homogeneous rotation/translation transformation $[R(t)|\mathbf{p}(t)]$ along the planned camera trajectory $\Gamma(t)$, the 2-D image feature $\mathbf{s}_j = [u_j v_j]^T$ of the 3-D target feature point $\mathcal{P}_j$ with homogeneous coordinates ${}^oP_j$ in the object frame is computed as

$$Z_j(u_j, v_j, 1)^T = A\big[R^T \,\big|\, -R^T\mathbf{p}\big]{}^oP_j \tag{4}$$

where Z_j is the depth of the target feature $\mathcal{P}_j$ in the camera frame. Similarly, all object features $\mathcal{P}_j$ are projected to their pixel coordinates $\mathbf{s}_j$ for $j = 1, \ldots, n$. The aforementioned projection is repeated for all camera poses along the planned camera trajectory to obtain the corresponding feature trajectories in the image.

IV. Alternate Exploration of Camera State Space and Robot Joint Space

Our proposed planning approach explores the camera state space by extending an exploring tree as in rapidly exploring random tree (RRT) approach [40] in the camera state space and simultaneously tracking the tree local paths in the robot joint space (see Fig. 4). Therefore, through this strategy, the search in the camera state space is used to effectively guide the search in the robot joint space.

As shown in Algorithm 1, the exploring tree $\mathcal{T}$ is iteratively extended in the camera state space until either it reaches the

Algorithm 1: Alternate camera state space and joint space exploration

input : Camera initial and desired states, $\mathbf{x}_i$ and $\mathbf{x}_d$
output: Camera trajectory Γ, or null in case of failure

1 **begin**
2 $\mathcal{T} \leftarrow$ `InitializeTree(`$\mathbf{x}_i$`);`
3 **repeat**
4 $\mathbf{x}_{rand} \leftarrow$ `GenerateRandomCameraState();`
5 $\mathbf{x}_{near} \leftarrow$ `FindNearestStateInTree(`$\mathcal{T}$, $\mathbf{x}_{rand}$`);`
6 $(\mathbf{x}_{new}, \mathcal{E}_x) \leftarrow$ `ExtendWithImageConstraints(`$\mathbf{x}_{near}$, $\mathbf{x}_{rand}$`);`
7 **if** $\mathcal{E}_x \neq$ null **then**
8 $(\mathbf{q}_{new}, \mathcal{E}_q) \leftarrow$ `TrackCameraPathInJointSpace(`$\mathcal{E}_x$`);`
9 **if** $\mathcal{E}_q \neq$ null **then**
10 Add $\mathbf{x}_{new}$ and $\mathcal{E}_x$ to tree $\mathcal{T}$;
11 **if** `rand()` $\leq p_{greedy}$ **then**
12 $(\mathbf{x}_d, \mathcal{E}_x) \leftarrow$ `ExtendGreedyToGoal(`$\mathbf{x}_{new}$, $\mathbf{x}_d$`);`
13 **if** $\mathcal{E}_x \neq$ null **then**
14 Add $\mathbf{x}_d$ and $\mathcal{E}_x$ to tree $\mathcal{T}$;
15 $\Gamma \leftarrow$ `RetrieveCameraTrajectory(`$\mathcal{T}$, $\mathbf{x}_d$`);`
16 **return** Γ;
17 **until** timeout;
18 **return** null ;

goal or the planning time is up. At each iteration, a node with a random camera state $\mathbf{x}_{\rm rand}$ is generated (Line 4) and the node with the nearest state $\mathbf{x}_{\rm near}$ to $\mathbf{x}_{\rm rand}$ in the tree is found (Line 5). Then, $\mathbf{x}_{\rm near}$ is extended toward $\mathbf{x}_{\rm rand}$, while considering the dynamics of the camera as a rigid body. The local path $\mathcal{E}_x$ obtained as the result of this extension is projected into the image space to check for image constraints (i.e., field of view limits and occlusions of target object by other obstacles or itself) by sampling synthetic images along the local path (Line 6). This kinodynamic extension of the tree is explained with more details in Section IV-A.

Given that the local camera path violates no image constraints, it is then tracked in the robot joint space using a local planner to check for physical constraints, i.e., collision with obstacles and joint limits (Line 8). We explain the local approach employed for joint space tracking in Section IV-B. In the case of successful tracking of the camera local path in joint space, the newly generated state $\mathbf{x}_{\rm new}$ (along with edge $\mathcal{E}_x$) is added as a new node to the camera tree (Line 10).

After each successful addition of a new node to the camera tree, with a probability $p_{\rm greedy}$, a greedy extension from the newly added node is attempted toward the goal state $\mathbf{x}_d$ (Line 12). The greedy extension approach has been detailed in Section IV-A.

We explain the details of camera tree expansion in its state space by referring to the corresponding line numbers in Algorithm 1.

A. Kinodynamic Planning in Camera State Space (Lines 4–6, Algorithm 1)

A randomized kinodynamic planning approach (similar to [41]) is employed as a local planner within our proposed planning framework to generate smooth camera trajectories.

Each node of the camera tree encodes a state vector $\mathbf{x}(t)$ of the camera as in (3). We model the camera as an unconstrained rigid body of mass M and body inertia I. At each extension of the camera tree, a random state vector $\mathbf{x}_{\rm rand}$ is generated in the camera state space (Line 4), and the tree node $\mathbf{x}_{\rm near}$ with the closest distance to $\mathbf{x}_{\rm rand}$ is found (Line 5) based on the following metric $\rho(\mathbf{x}_1, \mathbf{x}_2)$, which is defined as a measure of relative closeness between two states [41], e.g., $\mathbf{x}_1$ and $\mathbf{x}_2$

$$\rho(\mathbf{x}_1, \mathbf{x}_2) = w_p \|\mathbf{p}_1 - \mathbf{p}_2\| + w_h(1 - |\mathbf{h}_1 \cdot \mathbf{h}_2|) + w_v \|\mathbf{v}_1 - \mathbf{v}_2\| + w_\omega \|\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2\| \quad (5)$$

where $\mathbf{p}_1$ and $\mathbf{p}_2$ are the camera positions, $\mathbf{h}_1 \cdot \mathbf{h}_2$ denotes the inner product of unit quaternions corresponding to camera orientations, $\mathbf{v}_1$ and $\mathbf{v}_2$ are the camera linear velocities, and $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$ are the camera angular velocities pertinents to two states, $\mathbf{x}_1$ and $\mathbf{x}_2$, respectively. The weight coefficients w_p, w_h, w_v, and w_ω are user defined.

The camera tree is then extended from $\mathbf{x}_{\rm near}$ toward $\mathbf{x}_{\rm rand}$ by applying appropriate force $\mathbf{F}(t)$ and torque $\boldsymbol{\tau}(t)$ control inputs (Line 6). The equation of motion for the camera under the influence of control input $\mathbf{u}(t) = (\mathbf{F}(t), \boldsymbol{\tau}(t))$, i.e., the pair of force-torque input applied to the camera, is given as [42]

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) = \begin{bmatrix} \dot{\mathbf{p}}(t) \\ \dot{\mathbf{h}}(t) \\ \dot{\mathbf{v}}(t) \\ \dot{\boldsymbol{\omega}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{v}(t) \\ \frac{1}{2}\boldsymbol{\omega}_\times(t) \cdot \mathbf{h}(t) \\ \mathbf{F}(t)/M \\ R(t) I^{-1} R(t)^T \boldsymbol{\tau}(t) \end{bmatrix} \quad (6)$$

where $\boldsymbol{\omega}_\times(t) \cdot \mathbf{h}(t)$ denotes the quaternion dot product between $[0\ \omega_x\ \omega_y\ \omega_z]^T$ and $\mathbf{h}(t)$. The rotation matrix $R(t)$ and its transpose $R(t)^T$ are computed by converting the quaternion $\mathbf{h}(t)$ to its matrix representation.

In the literature pertinent to RRT (see, e.g., [41]), usually the input $\mathbf{u}$ is chosen at random, or selected by trying all the inputs from a discretized set of possible inputs and choosing the one that yields a new state closest to the randomly selected state $\mathbf{x}_{\rm rand}$. Note that this does not sacrifice the performance of the RRT extension since extension is still achieved toward the randomly selected node, and hence, the tree will be expanded into Voronoi region of the random node, thereby guaranteeing the extension of the tree into less explored regions of the space (see [41]). We have defined nominal intervals for the amount of force and torque from which the magnitudes of force and torque inputs are randomly selected for each extension, i.e., $\|\mathbf{F}(t)\| \in [0, F_{\max}]$ and $\|\boldsymbol{\tau}(t)\| \in [0, \tau_{\max}]$; and the directions of the force and torque inputs are computed along the relative direction of the position and orientation of the randomly selected state $\mathbf{x}_{\rm rand}$ with respect to the current state, i.e., $\overrightarrow{\mathbf{F}} = \frac{(\mathbf{p}_{\rm rand} - \mathbf{p}_{\rm near})}{\|\mathbf{p}_{\rm rand} - \mathbf{p}_{\rm near}\|}$, and $\overrightarrow{\boldsymbol{\tau}}$ is chosen as the axis of the quaternion that represents the relative orientation between $\mathbf{x}_{\rm near}$ and $\mathbf{x}_{\rm rand}$, which is computed as $\mathbf{h}_{\rm near}^{-1}\mathbf{h}_{\rm rand}$.

The camera tree is extended from $\mathbf{x}_{\rm near}$ by applying the control input $\mathbf{u}$ and integrating (6) over a user-defined time step Δt, which can be fixed (in our implementation, $\Delta t = 2$ s) or selected randomly at each iteration from a range of values $(0, \Delta t_{\max}]$.

The length of each extension has a direct relationship with Δt, which can be chosen arbitrarily (but nonzero) with the knowledge that higher values of Δt yield longer tree edges (with higher probability of violating a constraint along the extension in cluttered space) but faster extension of the tree in wide open regions, while lower values of Δt result in shorter edges, and slow but dense extension of the tree into the space. Usually, in a cluttered environment, lower time steps are recommended. The aforementioned integration requires a numerical approximation of (6). Given the current state $x(t)$ and input $\mathbf{u}(t)$ applied over a time interval Δt, the task is to compute the next state $\mathbf{x}(t+\Delta t)$. Since the dynamic model of the camera does not include contacts/collisions, the equations of motion are nonstiff, and we use a simple fixed-step Euler method for numerical integration, i.e., $\mathbf{x}(t+\delta_t) = \mathbf{x}(t) + \delta_t f(\mathbf{x}(t), \mathbf{u}(t))$, where δ_t is the integration time step (we used $\delta_t = 0.04$ s in our experiments).

When a camera subpath is extracted as previously, target object visibility checks are performed to see whether the target object is visible along the entire subpath (Line 6). Note that if a camera subpath is found to be feasible after checking for image constraints, the corresponding joint space subpath will be checked again (by a local planner) for joint limits and collision with the obstacles as detailed in Section IV-B.

1) Field of View Limits: The camera subpath should be examined for the field of view constraints. For this, we first use perspective projection to project object 3-D features into the image space using (4). For a 3-D point $\mathcal{P}_j$ with coordinates $[X_j\ Y_j\ Z_j]^T$ in the camera and image plane coordinates $\mathbf{s}_j = [u_j\ v_j]^T$, following simple inequalities check whether $\mathcal{P}_j$ remains in the camera field of view

$$\left.\begin{aligned} u_{\min} < u_j < u_{\max} \\ v_{\min} < v_j < v_{\max} \\ Z_j > 0 \end{aligned}\right\} \quad \text{for} \quad j = 1, \ldots, n \tag{7}$$

where $u_{\min}, u_{\max}, v_{\min}$, and $v_{\max}$ define the field of view limits, and n denotes the number of target features. The last constraint in (7) merely guarantees that the target features will remain in front of the camera.

2) Occlusion Due to Physical Obstacles: Target features' occlusions (due to workspace obstacles, robot body, or the target itself) for each camera pose along the camera subpath are checked by ray tracing from the camera optical center toward each target feature. This is done by checking for collision between the ray radiated from the point feature toward the camera optical center and the physical obstacles including workspace obstacles, robot body, and target itself. One could also employ the dynamic occlusion checking technique, which is proposed in [31], to check for occlusion along the camera subpath. However, the technique in [31] requires explicit computation of the boundary of occluded and visible regions.

B. Tracking Camera Local Paths in Joint Space (Line 8, Algorithm 1)

The camera subpaths generated in the state space are tracked in joint space using a resolved motion rate local controller as in [43]. Given a desired camera pose $\mathbf{r}_d(t)$ and velocity screw $\dot{\mathbf{r}}_d(t)$ along the camera subpath $\mathcal{E}_x$, the robot current configuration $\mathbf{q}(t)$ is advanced as

$$\mathbf{q}(t+d_t) = \mathbf{q}(t) + \dot{\mathbf{q}} d_t \tag{8}$$

with

$$\dot{\mathbf{q}} = K_p \widehat{\mathbf{J}}^{+} (\mathbf{r}_d(t) - \mathbf{r}(t)) + \widehat{\mathbf{J}}^{+} \dot{\mathbf{r}}_d(t) \tag{9}$$

where $\widehat{\mathbf{J}}^{+}$ is the pseudoinverse of a nominal model of the robot Jacobian $\mathbf{J} = \partial \mathbf{r} / \partial \mathbf{q}$, K_p is a positive proportional gain, and d_t is the integration time step. The new configuration is then checked for collision with obstacles and joint limits. Moreover, one could also check for singularities in robot Jacobian and joint velocity limits. After successful tracking of the whole camera subpath in joint space, it is then added to the tree along with its endpoint as the new node.

C. Greedy Extension of Camera Tree Toward Goal (Line 12, Algorithm 1)

Following each successful extension of the camera tree, a greedy extension is attempted toward the desired camera goal state $\mathbf{x}_d$, which is defined by the camera desired position and orientation, i.e., $\mathbf{p}_d$ and R_d, and zero linear and angular velocities, i.e., $\mathbf{v}_d = \mathbf{0}$ and $\omega_d = \mathbf{0}$.

The greedy extension of the camera tree can be formulated in the form of a boundary value problem, i.e., we are interested in computing a camera trajectory $\mathbf{x}(t)$ for $[t_0, t_1]$ from a given camera state $\mathbf{x}(t_0) = \mathbf{x}_0$ to the desired camera state $\mathbf{x}(t_1) = \mathbf{x}_d$ subjected to (6). The interval $t_g = t_1 - t_0$ is a predefined greedy extension time. To solve the aforementioned boundary value problem, one needs to design proper force and torque input profiles for the entire greedy extension interval. However, we devise a trajectory planning approach [44] by using cubic polynomials to generate camera position and orientation trajectories along the greedy extension. Therefore, the trajectories of camera position $\mathbf{p}(t)$ and orientation $\mathbf{h}(t)$ (represented as unit quaternion) are given as

$$\mathbf{p}(t) = A_3 t^3 + A_2 t^2 + A_1 t + A_0 \tag{10}$$

$$\mathbf{h}(t) = B_3 t^3 + B_2 t^2 + B_1 t + B_0 \tag{11}$$

for $t \in [t_0, t_1]$, where A_i's and B_i's are coefficient matrices that can be computed given the initial and end pose and velocity conditions imposed by $\mathbf{x}_0$ and $\mathbf{x}_d$, as well as the termination time.

Similar to other camera subpaths in the camera tree, the trajectory obtained using the aforementioned polynomials is then checked for image constraints and, in case of success, it is then tracked in the joint space.

Remarks on Complexity: In practice, sampling-based planning algorithms (including RRT-based planners as we adopted in this study) have been quite practically effective for high DOF spaces. RRT is probabilistically complete, i.e., the probability that a solution is found tends to 1, as the number of sampling/extension iterations approach infinity (see [41] for the proof). Unfortunately, as pointed out in [41], it remains a challenge to express the RRT convergence rate for a particular problem in terms of parameters that can be measured easily. As a

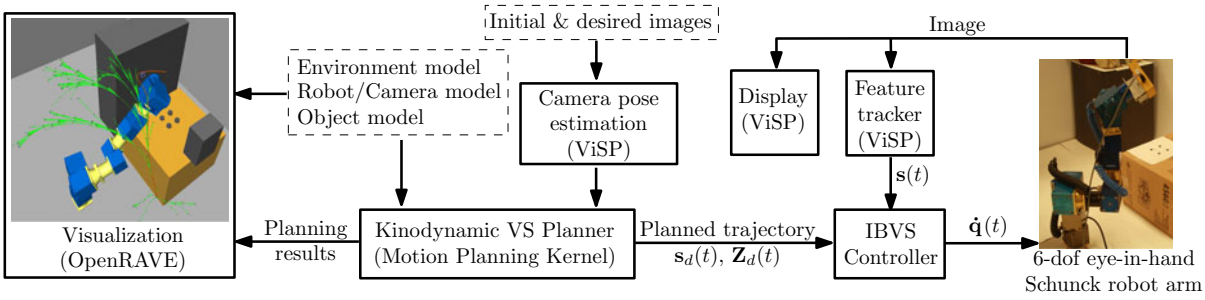


Fig. 5. Software/system architecture of our proposed framework.

common practice as well as to give the reader a better understanding of problem complexity, we have reported the planning time and the number of successfully explored nodes for each of the experiments that are provided in Section VI.

V. Implementation

The software/system architecture of our proposed framework is shown in Fig. 5. The proposed randomized kinodynamic planning approach has been implemented using our in-house motion planning kernel (MPK) [45]. The obstacles are imported as known CAD models into our planning environment. The MPK uses V/I-Collide library to perform collision checking given the objects CAD models. Note that the type of environment representation does not affect the planning scheme given that proper constraint checkers (e.g., collision and occlusion) are in place for the specific representation of the environment. For instance, in a future extension of our study, we can easily imagine that the environment model can be acquired using perception techniques based on stereo/depth sensors. We used the camera factory settings (i.e., intrinsic parameters) with no further calibration.

The initial and desired camera poses are computed using Dementhon's technique [38], and the virtual visual servoing approach is provided through the visual servoing platform (ViSP) [46]. For visualization purposes, the planning results, which include the planned robot/camera trajectory and the camera tree, are sent to an OpenRAVE [47] viewer plug-in where the robot trajectory is visualized/animated. The planned feature trajectories are sent to an IBVS controller that generates appropriate robot/camera velocity inputs to our 6-DOF *Schunck* robotic arm with an in-hand camera. We employed the dot feature tracker in ViSP to keep tracking of multiple black markers on the target object. The IBVS controller is explained in more details as follows.

A. Tracking Feature Trajectories Using Image-Based Visual Servoing

We adopt an image-based trajectory tracking controller as in [9] to track the desired feature trajectories, $\mathbf{s}_d(t)$; the following error function is defined in the image space:

$$\mathbf{e}_s = \mathbf{s}(\mathbf{r}(\mathbf{q}(t))) - \mathbf{s}_d(t) \tag{12}$$

where $\mathbf{s}(\mathbf{r}(\mathbf{q}(t)))$ is the vector of current image features at the current camera pose $\mathbf{r}$ specified by the robot configuration $\mathbf{q}(t)$.

Taking the derivative of (12) with respect to time, we have

$$\begin{aligned}\dot{\mathbf{e}}_s &= \frac{\partial \mathbf{s}}{\partial \mathbf{r}}\frac{\partial \mathbf{r}}{\partial \mathbf{q}}\dot{\mathbf{q}} - \dot{\mathbf{s}}_d \\ &= \mathbf{L}\mathbf{J}\dot{\mathbf{q}} - \dot{\mathbf{s}}_d \end{aligned} \tag{13}$$

where $\mathbf{L} = \frac{\partial \mathbf{s}}{\partial \mathbf{r}}$ denotes the image Jacobian, and $\mathbf{J} = \frac{\partial \mathbf{r}}{\partial \mathbf{q}}$ is the robot Jacobian. To impose exponential decay of $\mathbf{e}_s(t)$ to zero, we let $\dot{\mathbf{e}}_s(t) = -\lambda_s \mathbf{e}_s$, where λ_s is a positive gain. Hence, we have

$$\dot{\mathbf{q}} = (\mathbf{L}\mathbf{J})^+(-\lambda_s \mathbf{e}_s + \dot{\mathbf{s}}_d) \tag{14}$$

where $\dot{\mathbf{q}}$ denotes the joints velocity input, and $(.)^+$ denotes the Moore–Penrose pseudoinverse operation. For a 3-D point $\mathcal{P}_j$ with coordinates $[X_j\ Y_j\ Z_j]^T$ in the camera coordinate frame and with image pixel coordinates $\mathbf{s}_j = [u_j\ v_j]^T$ where $[u_j\ v_j 1]^T = A[x_j\ y_j\ 1]^T$, the interaction matrix, which is related to $\mathbf{s}_j$, is given as

$$L(\mathbf{s}_j, Z_j) = \begin{bmatrix} fp_u & 0 \\ 0 & fp_v \end{bmatrix} \cdot \begin{bmatrix} -\frac{1}{Z_j} & 0 & \frac{x_j}{Z_j} & x_j y_j & -1 + x_j^2 & y_j \\ 0 & -\frac{1}{Z_j} & \frac{y_j}{Z_j} & 1 + y_j^2 & -x_j y_j & -x_j \end{bmatrix} \tag{15}$$

where $x_j = (u_j - u_0)/fp_u$, and $y_j = (v_j - v_0)/fp_v$. The interaction matrix for n feature points $\mathbf{s} = [\mathbf{s}_1^T, \ldots, \mathbf{s}_n^T]^T$ with corresponding depths $\mathbf{Z} = [Z_1, \ldots, Z_n]^T$ can then be created by stacking up the interaction matrices of all features as

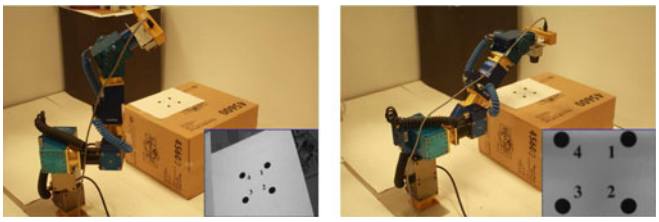
$$\mathbf{L}(\mathbf{s}, \mathbf{Z}) = \left[L^T(\mathbf{s}_1, Z_1) \ldots L^T(\mathbf{s}_n, Z_n)\right]^T. \tag{16}$$

As it is seen, the interaction matrix depends on the depth information, which is not usually available at execution time. Hence, a nominal value is used instead [48]. Similar to [9], we use $\widehat{\mathbf{L}} = \mathbf{L}(\mathbf{s}_d(t), \mathbf{Z}_d(t))$ as a nominal model of the interaction matrix that is created by stacking up the interaction matrices related to all the current desired image features along the planned trajectory. Moreover, a precise model of the robot kinematics may not be available in practice. Hence, a nominal robot Jacobian $\widehat{\mathbf{J}}$ is used in the formulation of the control law.

Finally, the control law can be written as

$$\dot{\mathbf{q}} = (\widehat{\mathbf{L}}\widehat{\mathbf{J}})^+(-\lambda_s \mathbf{e}_s + \dot{\mathbf{s}}_d). \tag{17}$$

In the following section, first, we present results of our experiments that demonstrate the effectiveness of the proposed



(a) (b)

Fig. 6. Field of view and joint limits avoidance experimental setup. (a) Start configuration of the robot. (b) Goal configuration. Target features are the four black dots on the white paper on the box. Inset shows the features in the respective images at start and goal.

planning framework in accounting for image and physical constraints in visual servoing tasks.

VI. Experimental Results

We validated the effectiveness of the proposed planning framework on a 6-DOF eye-in-hand robotic arm. The target object is composed of black dots, as shown in Fig. 6. For ease of image processing, black dots on white background were used as features; however, one could easily substitute this step with more general features, such as lines and contours, as long as real-time tracking is available for features of interest.

The focus of these experiments is to demonstrate the capabilities of the proposed approach in taking physical and image constraints into account. To better visualize and illustrate the effectiveness of our approach, we first show how it takes care of field of view and joint limit constraints, and then show how it takes care of occlusion and obstacle avoidance constraints in a cluttered workspace. We contrast our results with an IBVS method in which the control law is given as

$$\dot{\mathbf{q}} = -\lambda_s (\widehat{\mathbf{L}}\widehat{\mathbf{J}})^{+} (\mathbf{s}(t) - \mathbf{s}^*) \tag{18}$$

where $\mathbf{s}^* = \mathbf{s}_d(t_f)$ contains the fixed final desired features, and the interaction matrix $\widehat{\mathbf{L}}$ is computed at $(\mathbf{s}^*, \mathbf{Z}^*)$.

In all experiments, we used the camera intrinsic parameters provided by the manufacture with no further calibration. The video clips of these experiments are available online at https://sites.google.com/site/moslemk/research/kinodynamic-planning-for-vs

A. Field of View and Joint Limits Avoidance

The robot at the initial and desired configurations is shown in Fig. 6(a) and (b) along with the corresponding initial and desired images at their right-bottom corners. The corresponding camera displacement between the initial and desired camera frames is $[20\ 35\ 21]^T$ cm in *xyz* translation and $[34\ 26\ 8]^T$ degrees in roll–pitch–yaw rotation. Fig. 7 shows the results obtained using the IBVS technique based on the control law in (18). As seen in Fig. 7(a), some of the feature trajectories (features 1 and 4 specifically) leave the camera's field of view. Fig. 7(b) shows the corresponding joint trajectories. All joint values have been normalized to $[-1, 1]$ where 1 and -1 determine the upper and lower limits for each joint, respectively. As shown in Fig. 7(b), the trajectories for joints 2 and 3 violate their limits. Either of the aforementioned violations results in failure of the visual servoing task.

We applied our proposed approach to the same task and the results are presented in Fig. 8. Using our proposed approach, we planned feasible feature trajectories in the image space, as shown in Fig. 8(a). These trajectories do not violate the camera field of view and robot joint limits that have been taken into account at the planning stage. The robot has been servo controlled along the planned feature trajectories using the tracking control law given in (14). As shown in Fig. 8(b), the followed feature trajectories respect the camera field of view, and remain within the image limits. Fig. 8(e) shows the corresponding joint trajectories that are within their respective limits. The planned camera tree (in green), along with the camera trajectory (in red), are shown in Fig. 8(h). The solution trajectory was found in about 80 s after about 1500 states were successfully explored. Note that the transient tracking errors in image space [see Fig. 8(c)] are initially large (up to 30 pixels), primarily due to image measurement errors in segmenting the centers of 2-D features, and slow tracking control loop (12 Hz) due to the limited frame rate (15 Hz) and delays in processing the image. Supported by our empirical results presented in Section VII, as well as existing literature [9], we believe that a faster vision system ($\approx$50 Hz, as studied in [37]) will highly improve the tracking performance in the image space.

B. Obstacle Collision and Occlusion Avoidance

In the following experiments, the robot workspace is cluttered with obstacles (i.e., boxes), which make some parts of the workspace unreachable by the robot (see Fig. 9). The obstacles also create occluded regions in the workspace from which the target object is occluded and not visible in the camera image. The robot, at its initial and desired configurations, along with the corresponding images taken by the camera, is shown in Fig. 9.

First, we employed the pure IBVS technique to carry out the task of servoing the robot toward its desired configuration (see Fig. 10). The robot trajectory induced by the IBVS technique results in both occlusion of the target object (see Fig. 10, left bottom) and robot collision with workspace obstacles (see Fig. 10). Hence, the IBVS technique failed to accomplish this task and was stopped. Most of the previous approaches on path planning for visual servoing (as briefly reviewed in Section I and with more details in [4]) either failed to incorporate general occlusion and collision avoidance constraints or were susceptible to local minima in such complex environments.

Our proposed randomized planning framework can, however, effectively take care of these constraints and, hence, ensures robust execution of the servoing task with respect to the violation of image and physical constraints. The planned camera tree along with the camera trajectory has been visualized in Fig. 11(a). The solution trajectory was found in about 200 s after about 900 states were successfully explored. The corresponding feature trajectories, which are obtained by projecting the planned camera trajectory into the image space, are shown in Fig. 11. The robot, when servo controlled along the desired

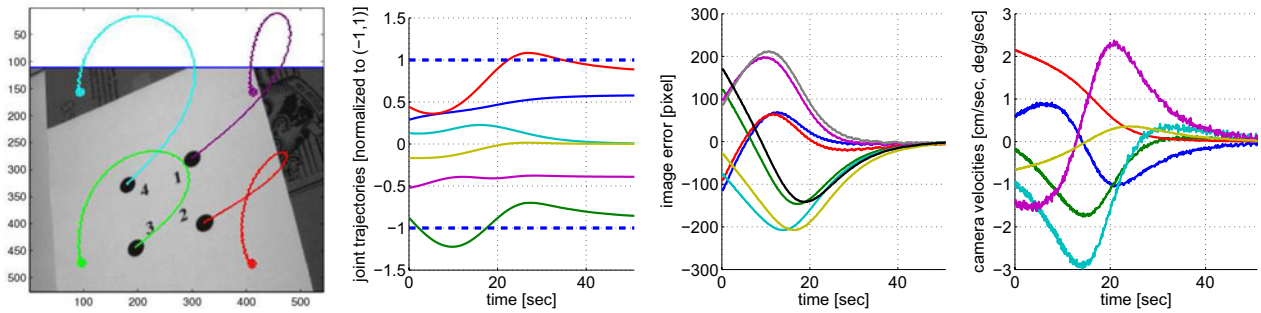


Fig. 7. Results obtained using pure IBVS. (a) Fature trajectories, (b) joint trajectories, (c) tracking error in image space, and (d) camera velocity inputs.

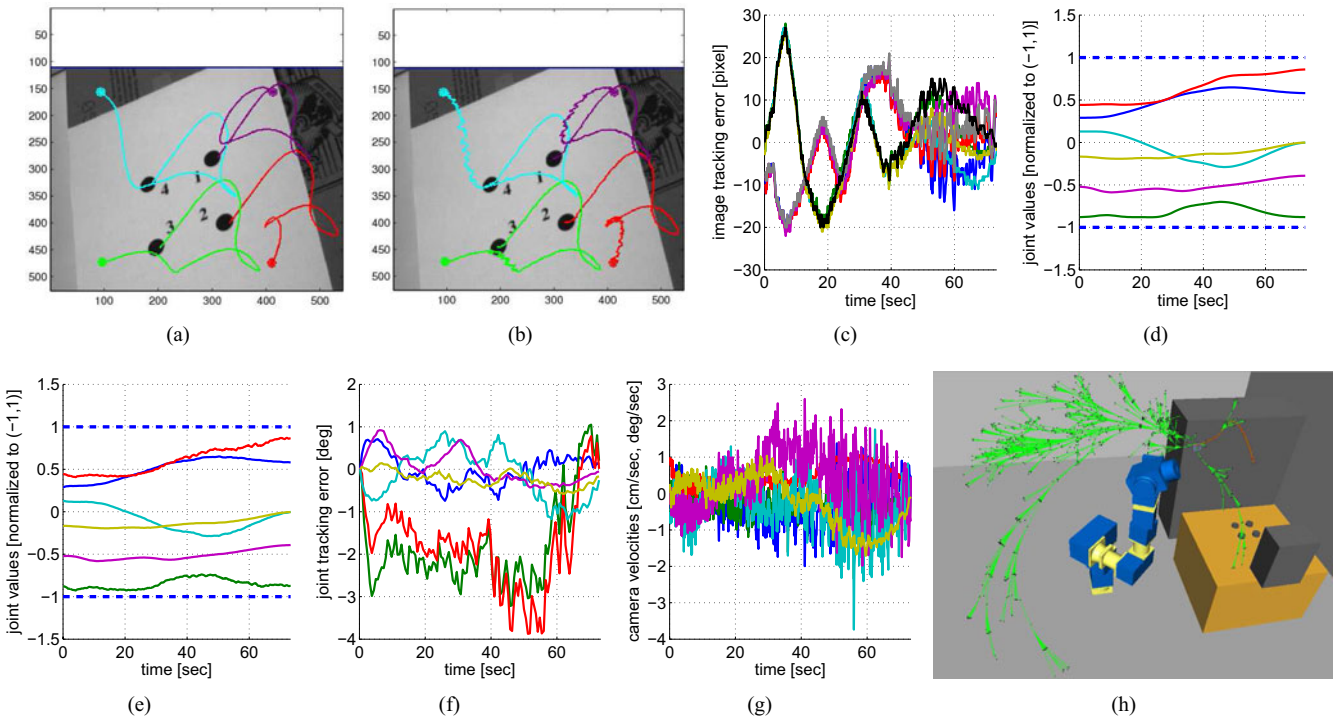


Fig. 8. Results obtained using our proposed kinodynamic planning approach in presence of field of view and joint limits. (a) Planned feature trajectories, (b) followed feature trajectories, (c) tracking error in image space, (d) planned joint trajectories, (e) followed joint trajectories, (f) tracking error in joint space, (g) camera velocity inputs, and (h) planned camera tree (in green) and the solution trajectory (in red).

feature trajectories, reaches its desired configuration, avoiding occlusion of any of the object features, while avoiding collision with workspace obstacles. The feature trajectories followed by the servoing control law (14) are shown in Fig. 11(c). The corresponding planned and followed robot joint trajectories are shown in Fig. 11(e) and (f), respectively.

As is apparent from Fig. 11(g), the joint tracking error of up to about 9° is somewhat more than desired, and a systematic way to reduce this error would be via a new controller that regulates errors in both image and joint spaces, which is indeed a challenging task and future work.

VII. Discussion

As mentioned earlier, in this study, the planning is performed without accounting for uncertainties in modeling and calibration. Although we assume that the system is nominally calibrated and we have access to robot model and camera parameters provided by the manufacturer, the effect of uncertainty may not be fully removed, and the tracking performance is highly affected as noticed through the results presented in the previous

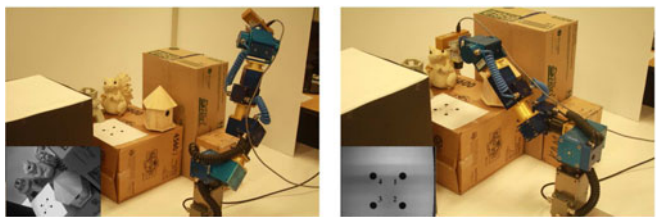


Fig. 9. Experimental setup for the collision and occlusion avoidance experiment. In this experiment, the robot has to be servo controlled toward its desired configuration located in a narrow empty space between the obstacles that cause occlusion of the target and/or collision with the robot.

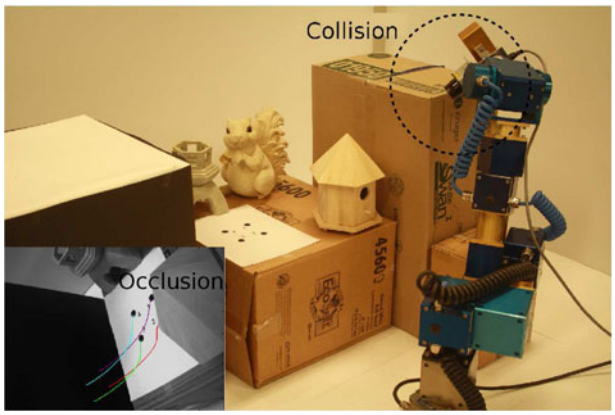


Fig. 10. Results obtained using pure IBVS. The trajectories induced by the IBVS technique result in occlusion of some of the target features by the workspace obstacles (see the image at the bottom-left corner), and collision of the robot with workspace obstacles as shown at the top-right corner. Hence, the servoing task fails.

section. We derive the error dynamics for an IBVS controller (controller I) and for a joint space controller (controller II). However, a definitive general statement and a thorough analytical study of parameters' uncertainty and their effects on errors in image and joint spaces is not apparent from these expressions. Therefore, we also provide empirical results on the effects of uncertainty on tracking errors in image and joint spaces for both controllers, and draw some empirical conclusions discussing the relative merits of the two controllers.

A. Image Space Trajectory Tracking (Controller I)

Error analysis: Replacing (17) in (13), we obtain the image error dynamics

$$\dot{\mathbf{e}}_s = -\lambda_s \mathbf{L}\mathbf{J}(\widehat{\mathbf{L}}\widehat{\mathbf{J}})^{+}\mathbf{e}_s + (\mathbf{L}\mathbf{J}(\widehat{\mathbf{L}}\widehat{\mathbf{J}})^{+} - \mathbf{I})\dot{\mathbf{s}}_d. \tag{19}$$

The error $\mathbf{e}_s$ will have an exponential decay if the term $\mathbf{L}\mathbf{J}(\widehat{\mathbf{L}}\widehat{\mathbf{J}})^{+}$ remains positive definite along the trajectory, which can be ensured locally, as long as the camera is controlled closely along the desired feature trajectories. However, the error will converge to zero only if either $\mathbf{L}\mathbf{J}(\widehat{\mathbf{L}}\widehat{\mathbf{J}})^{+} = \mathbf{I}$ or $\dot{\mathbf{s}}_d = 0$, which nullify the last term in (19). As also studied by Malis [2], setting a high gain λ_s or reducing the desired velocity $\dot{\mathbf{s}}_d$ will reduce the tracking error. We point out that, ensured by our kinodynamic planning approach, the desired velocity $\dot{\mathbf{s}}_d$ approaches zero, as the camera reaches its final desired pose $\mathbf{x}_f$, reducing the tracking error toward the end of the trajectory. Malis and Rives [49] have also studied the stability of controller I and showed that it is fairly stable in the presence of camera calibration errors, but with a smaller stability domain when coarse features' depth distribution is used.

B. Joint Space Trajectory Tracking (Controller II)

We start by deriving the joint trajectory tracking control law with the objective of tracking a desired (planned) trajectory $\mathbf{q}_d(t)$. The error function in joint space is defined as

$$\mathbf{e}_q(t) = \mathbf{q}(t) - \mathbf{q}_d(t) \tag{20}$$

and we obtain

$$\dot{\mathbf{e}}_q = \dot{\mathbf{q}}(t) - \dot{\mathbf{q}}_d(t). \tag{21}$$

To ensure exponential decay of the error to zero, we impose $\dot{\mathbf{e}}_q = -\lambda_q \mathbf{e}_q$ in the aforementioned equation, which yields the control law

$$\dot{\mathbf{q}} = -\lambda_q \mathbf{e}_q + \dot{\mathbf{q}}_d. \tag{22}$$

Error analysis: By closing the loop, i.e., replacing (22) in (21), the control law in (22) guarantees exponential convergence of error to zero in joint space, i.e.,

$$\dot{\mathbf{e}}_q = -\lambda_q \mathbf{e}_q. \tag{23}$$

C. Empirical Study of Calibration and Modeling Uncertainties

We provide an empirical comparison between the performance of the two aforementioned controllers under various conditions: uncertainties in camera focal length and robot kinematics, and image measurement errors. We consider a simple planning example for the sake of simulations, where the camera trajectory is defined as a 90° rotation around its optical axis. The target consists of the vertices of a square with the optical axis of the camera passing through its center and orthogonal to its plane. The camera is assumed to be mounted at the end-effector of a 6-DOF robotic arm (the one we used for our experiments in the previous section) in a way that the 90° rotation of the camera can be achieved by rotating joint 6 only. The desired feature and joint trajectories are shown in Fig. 12. The total trajectory running time is 30 s. We use controllers I and II to execute these trajectories and compare their performance. Both controllers are simulated at 50 Hz and have access to joint encoder data and image measurements at the same rate.

Ideal conditions: First, as a reference, we simulate an ideal condition, i.e., no calibration/modeling uncertainty and away from singularities in robot-image Jacobian. Under such ideal conditions, both controllers demonstrate fairly similar performances as shown through the tracking errors in both image and joint spaces in Fig. 13. These results serve as a reference for comparing the performance of controllers I and II under the following conditions simulated individually:

1) 5% error in camera focal length (see Fig. 14);
2) up to 5 pixels (random) image measurement error (see Fig. 15);
3) 1-cm position error, and 1° orientation error in robot forward kinematic model (see Fig. 16).

Summary of results: Comparing the tracking errors obtained as the result of using controller I (i.e., image trajectory tracking) with those yielded through controller II (i.e., joint trajectory tracking), a somewhat expected trend is observed for all uncertainties. These results do confirm our intuitive expectation that neither the image trajectory tracking nor the joint trajectory tracking can achieve low errors in both image and joint spaces. However, relatively speaking, the image trajectory tracking performs well (i.e., yielding an exponential decay of tracking error to zero) in image space [see (a) in Fig. 14–16], while the joint trajectories converge to within about 5° (i.e., not too large) steady-state error [see (b) in Fig. 14–16]. Joint trajectory

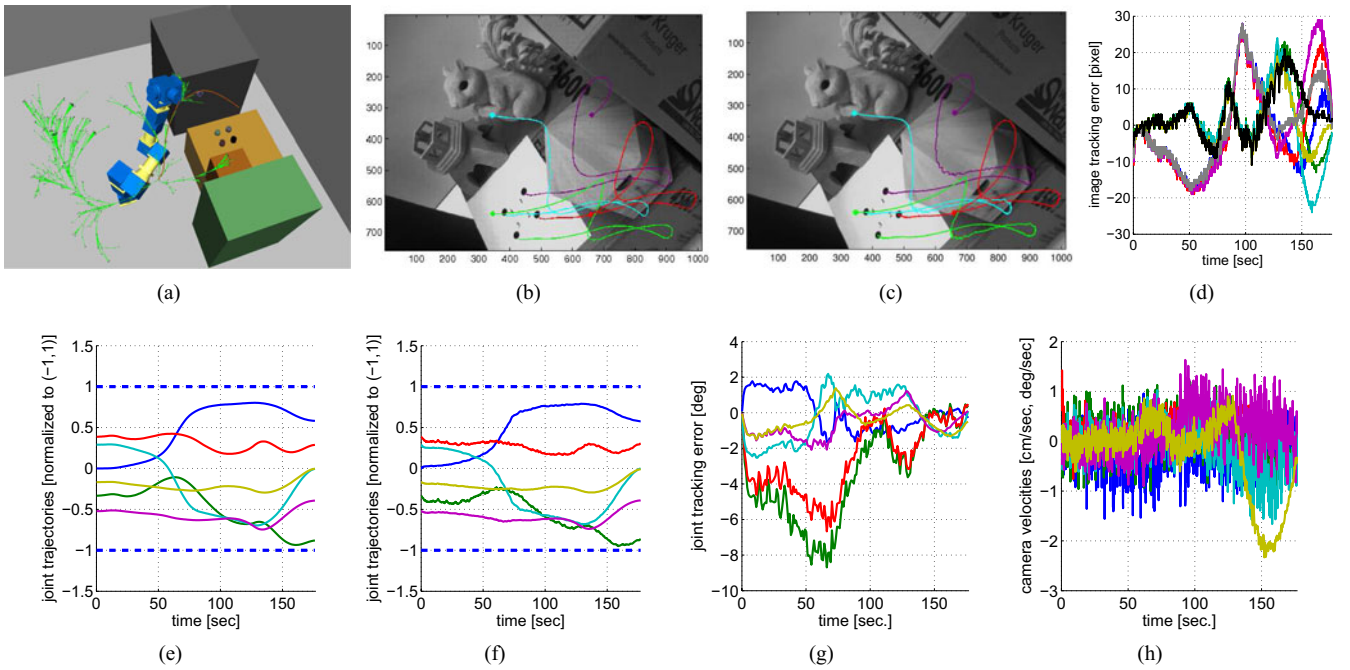


Fig. 11. Results obtained using the proposed kinodynamic planning approach. The planned camera tree (in green) along the planned trajectory (in red) is shown in (a). The robot (servo controlled along the planned feature trajectories) moves toward its desired configuration located in the narrow empty space between the obstacles, while avoiding collision with obstacles and keeping target in the field of view. Therefore, the servoing task has been performed successfully, (b) planned feature trajectories, (c) followed feature trajectories, (d) tracking error in image space, (e) planned joint trajectories, (f) followed joint trajectories, (g) tracking error in joint space, and (h) camera velocity inputs.

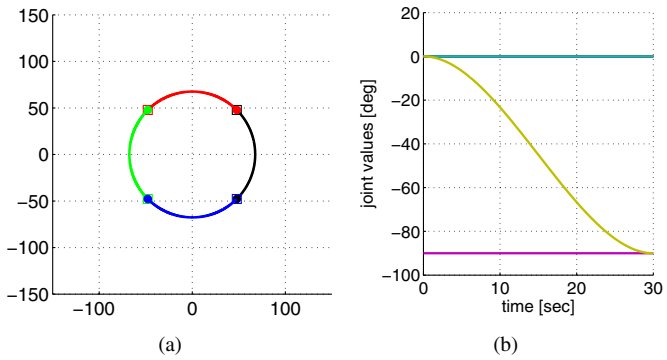


Fig. 12. (a) Desired trajectories pertinent to four point features. Each feature moves on a circular arc about 90° around the camera optical center. (b) Desired joint trajectories related to the feature trajectories in (a).

tracking yields an exponential decay of tracking error to zero in joint space [see (d) in Fig. 14–16] but with poor performance in image space [see (c) in Fig. 14–16], particularly in presence of forward kinematic errors (very large steady-state image errors of up to about 50 pixels). This lends some support to our decision to adopt an image space trajectory tracking controller.

We also note that the answer to the question of which controller should be used to execute the planned trajectories also depends on the task. That is, for example, if the task objective requires a close tracking of feature trajectories within the image while the error in joint space can be tolerated, then the image trajectory tracking controller should be adopted. For example, in MEMS micromanipulation applications, close tracking of microparts trajectories is very crucial to avoid occlusion of microparts by the gripper or losing them from the field of view. Hence, an image-based trajectory tracking throughout the whole trajectory is advisable (e.g., as adopted in [50]).

This study does not account for calibration, modeling errors, as well as sensing and control errors that may result in deviations from planned (joint space) paths during execution—such as the observed joint space tracking errors in our simulations and in experiments. Such deviations may result in violation of, for example, physical constraints (joint limits and collisions). However, we argue that it is indeed possible to extend the planning framework to incorporate such errors. We outline such an approach.

In order to account for such joint errors at planning stage, we need to know the error bounds as a function of uncertainty parameters. In the absence of any analytical results on these bounds so far, the key observation is that the joint space tracking errors can be estimated via a simulated image-based trajectory tracking step. More precisely, after each successful extension of the camera tree, the robot is simulated to move along the (locally) extended image trajectory using the same controller, as will be used in the actual execution (say, controller I, as explained previously) with appropriate uncertainties on kinematic model and/or camera parameters and, thus, determining the maximum deviation of joints from the planned joint trajectory. Once this error bound is known, we can easily determine the volume swept by the robot corresponding to the error bound (see the orientation slicing method in [19, Sec. 5.1, p. 283]), and augment the robot size by this swept volume. The RRT extension is then rechecked for collisions with the augmented robot and discarded if there is collision, and kept otherwise.

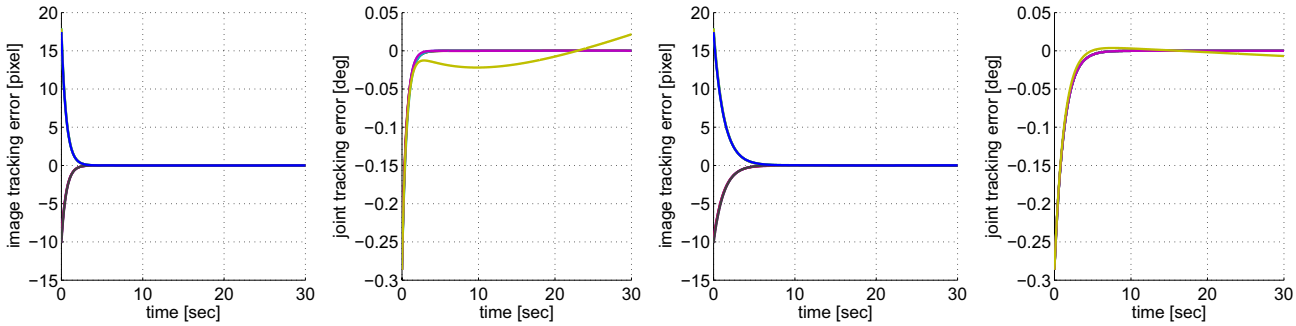


Fig. 13. Tracking errors in image and joint spaces under ideal conditions, i.e., no calibration/modeling uncertainties and away from singularities in robot-image Jacobian. (a) and (b) Image trajectory tracking. (c) and (d) Joint trajectory tracking.

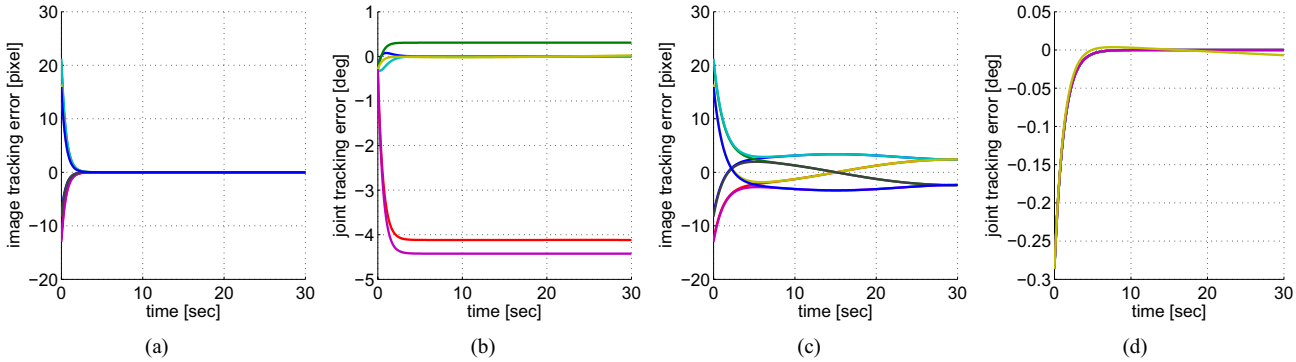


Fig. 14. Tracking errors in image and joint spaces in presence of 5% error in camera focal length. (a) and (b) Image trajectory tracking. (c) and (d) Joint trajectory tracking.

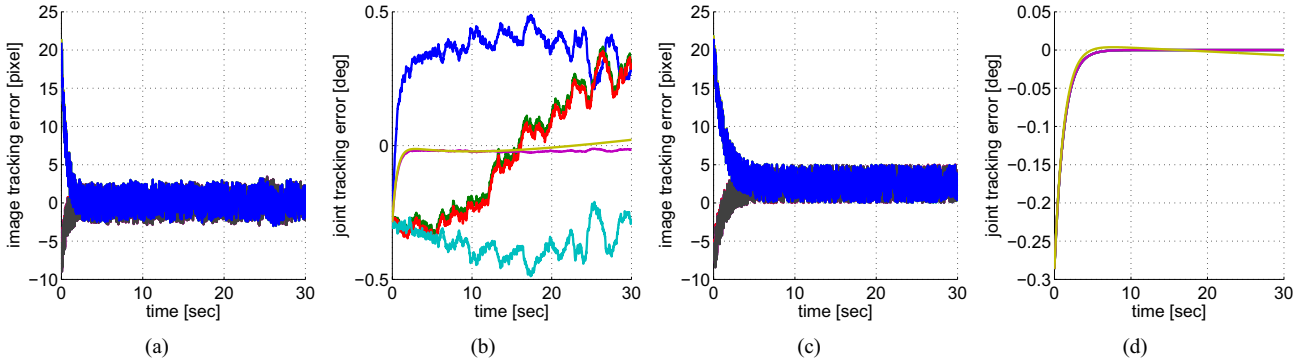


Fig. 15. Tracking errors in image and joint spaces in presence of up to 5 pixels (random) image measurement errors. (a) and (b) Image trajectory tracking. (c) and (d) Joint trajectory tracking.

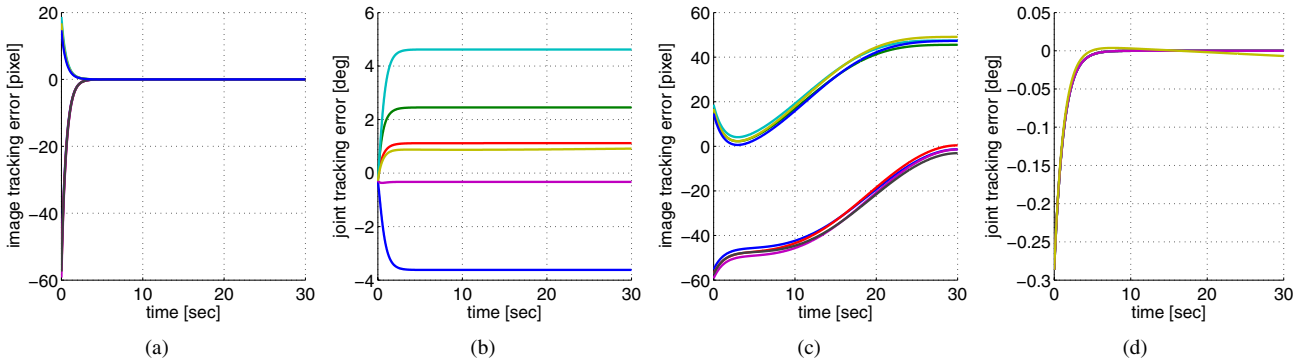


Fig. 16. Tracking errors in image and joint spaces in presence of 1 cm/deg error in a robot forward kinematic model. (a) and (b) Image trajectory tracking. (c) and (d) Joint trajectory tracking.

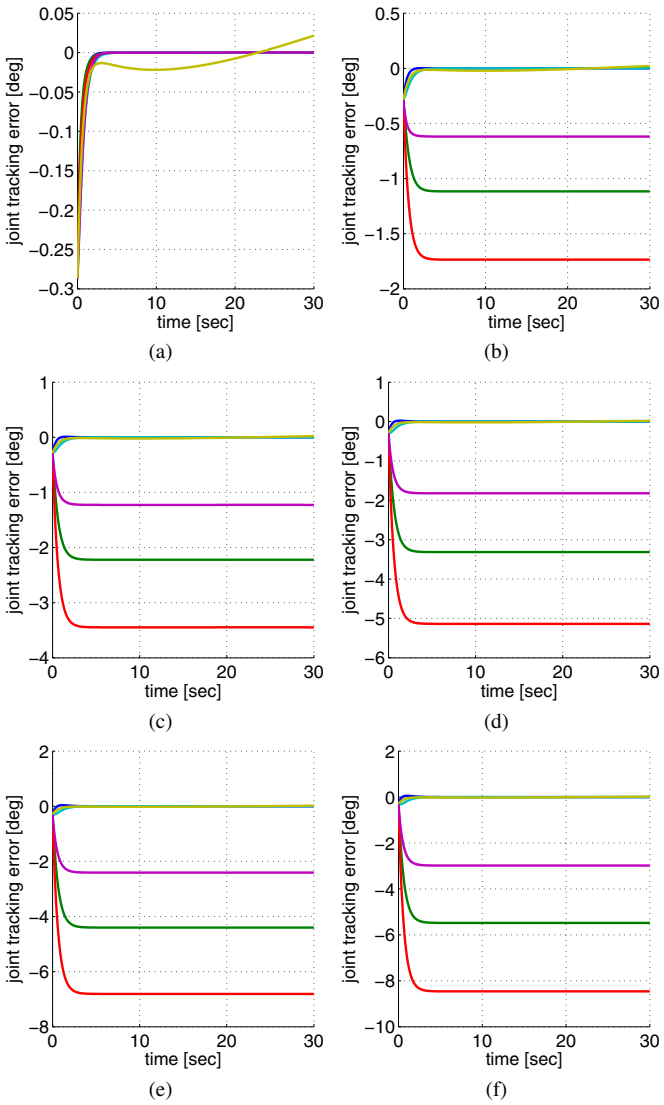


Fig. 17. Joint tracking errors obtained by applying an image-based trajectory tracking along the image trajectories in Fig. 12(a) under (a) 0%, (b) 1%, (c) 2%, (d) 3%, (e) 4%, and (f) 5% error on focal length. As shown, the joint tracking errors monotonically increase (or decrease) as the focal length error increases (or decreases).

However, one should note that the problem, in general, may not be convex, i.e., extreme levels of uncertainty in parameters may not correspond to extreme joint tracking errors. This makes determining joint error bounds while accounting for all uncertainties computationally intensive. For instance, if there were n uncertainty parameters, each discretized to k quantization levels, one would need to run the simulated control k^n times, and, hence, determine the global bounds for the joint errors. An approximate and faster alternative would be to consider only a subset of most influential parameters, e.g., camera intrinsic parameters such as camera focal length.

We carried out some simulations, allowing uncertainty in the camera focal length only. Fig. 17 shows the joint space tracking errors obtained for servoing the robot along the image trajectories in Fig. 12(a), under different amounts of uncertainty in focal length. As shown, the joint tracking errors monotonically increase (or decrease) as the focal length error increases (or decreases), i.e., extremes of the joint errors occur at extreme values of uncertainty. This convexity property can be utilized toward providing general guidelines for selecting the joint error bounds at the planning stage. Nonetheless, we point out that this property may not always hold, especially when including more than one source of uncertainty; hence, one needs to calculate more conservative bounds by simulating over the entire parameters' uncertainty space.

VIII. Conclusion

We have incorporated a randomized kinodynamic planning approach with image-based control of a robotic arm for a servoing task. The proposed approach yields planned C^1-smooth camera trajectories by taking camera dynamics into account, while accounting for a critical set of physical and image constraints: 1) maintaining continuous visibility of the target within the camera field of view; 2) avoiding visual occlusion of target features caused by the workspace obstacles, robot's body, or the target itself; 3) avoiding collision with physical obstacles or self-collision; and 4) joint limits. Incorporating these constraints at the planning stage increases the overall robustness of servoing tasks, with respect to the violation of the aforementioned constraints. The proposed framework is general and, therefore, can incorporate other constraints depending on the task at hand.

We are currently extending the proposed approach to a mobile manipulator system, i.e., a nonholonomic mobile base with an on-board 6-DOF robotic arm [51]. Two avenues are also sought to enhance this study: 1) extending this study toward (partially) uncalibrated domains by accounting for modeling and calibration uncertainties at the planning stage, and 2) designing a controller that can reconcile tracking errors in both image and joint spaces.

References

[1] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Trans. Robot. Autom.*, vol. 12, no. 5, pp. 651–70, Oct. 1996.

[2] E. Malis, "Visual servoing invariant to changes in camera-intrinsic parameters," *IEEE Trans. Robot. Autom.*, vol. 20, no. 1, pp. 72–81, Feb. 2004.

[3] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," in *The Confluence of Vision and Control*, D. Kriegman, G. Hager, and A. Morse, Eds. Berlin, Germany: Springer-Verlag, 1998, pp. 66–78.

[4] M. Kazemi, K. Gupta, and M. Mehrandezh, "Path planning for visual servoing: A review and issues," in *Visual Servoing via Advanced Numerical Methods*, G. Chesi and K. Hashimoto, Eds. Berlin, Germany: Springer, Mar. 2010, ch. 11, pp. 189–208.

[5] E. Malis, F. Chaumette, and S. Boudet, "2-1/2-D visual servoing," *IEEE Trans. Robot. Autom.*, vol. 15, no. 2, pp. 238–250, Apr. 1999.

[6] N. Gans, S. Hutchinson, and P. Corke, "Performance tests for visual servo control systems, with application to partitioned approaches to visual servo control," *Int. J. Robot. Res.*, vol. 22, nos. 10–11, pp. 955–981, 2003.

[7] L. Deng, F. Janabi-Sharifi, and W. Wilson, "Hybrid motion control and planning strategies for visual servoing," *IEEE Trans. Ind. Electron.*, vol. 52, no. 4, pp. 1024–1040, Aug. 2005.

[8] N. Gans and S. Hutchinson, "Stable visual servoing through hybrid switched-system control," *IEEE Trans. Robot.*, vol. 23, no. 3, pp. 530–540, Jun. 2007.

[9] Y. Mezouar and F. Chaumette, "Path planning for robust image-based control," *IEEE Trans. Robot. Autom.*, vol. 18, no. 4, pp. 534–549, Aug. 2002.

[10] M. Kazemi, K. Gupta, and M. Mehrandezh, "Kinodynamic planning for visual servoing," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 2478–2484.

[11] K. Hosoda, K. Sakamoto, and M. Asada, "Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3d reconstruction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Aug. 1995, vol. 1, pp. 29–34.
[12] J. Park and M. Chung, "Path planning with uncalibrated stereo rig for image-based visual servoing under large pose discrepancy," *IEEE Trans. Robot. Autom.*, vol. 19, no. 2, pp. 250–258, Apr. 2003.
[13] Y. Mezouar and F. Chaumette, "Optimal camera trajectory with image-based control," *Int. J. Robot. Res.*, vol. 22, nos. 10–11, pp. 781–803, 2003.
[14] F. Schramm and G. Morel, "Ensuring visibility in calibration-free path planning for image-based visual servoing," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 848–854, Aug. 2006.
[15] V. Kyrki, D. Kragic, and H. Christensen, "New shortest-path approaches to visual servoing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep.–Oct. 2004, vol. 1, pp. 349–354.
[16] B. Allotta and D. Fioravanti, "3D motion planning for image-based visual servoing tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 2173–2178.
[17] J. Borgstadt and N. Ferrier, "Visual servoing: Path interpolation by homography decomposition," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2001, vol. 1, pp. 723–730.
[18] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot. Res.*, vol. 5, no. 1, pp. 90–98, 1986.
[19] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA, USA: Kluwer, 1991.
[20] N. Cowan, J. Weingarten, and D. Koditschek, "Visual servoing via navigation functions," *IEEE Trans. Robot. Autom.*, vol. 18, no. 4, pp. 521–533, Aug. 2002.
[21] G. Chesi, D. Prattichizzo, and A. Vicino, "Straight line path-planning in visual servoing," *Trans. ASME, J. Dyn. Syst. Meas. Control*, vol. 129, no. 4, pp. 541–543, 2007.
[22] G. Chesi and Y. Hung, "Global path-planning for constrained and optimal visual servoing," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1050–1060, Oct. 2007.
[23] A. H. A. Hafez, A. K. Nelakanti, and C. V. Jawahar, "Path planning approach to visual servoing with feature visibility constraints: A convex optimization based solution," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct.–Nov. 2007, pp. 1981–1986.
[24] G. Chesi, "Visual servoing path planning via homogeneous forms and LMI optimizations," *IEEE Trans. Robot.*, vol. 25, no. 2, pp. 281–291, Apr. 2009.
[25] H. Zhang and J. Ostrowski, "Visual motion planning for mobile robots," *IEEE Trans. Robot. Autom.*, vol. 18, no. 2, pp. 199–208, Apr. 2002.
[26] S. Bhattacharya, R. Murrieta-Cid, and S. Hutchinson, "Optimal paths for landmark-based navigation by differential-drive vehicles with field-of-view constraints," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 47–59, Feb. 2007.
[27] G. Lopez-Nicolas, S. Bhattacharya, J. Guerrero, C. Sagues, and S. Hutchinson, "Switched homography-based visual control of differential drive vehicles with field-of-view constraints," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 4238–4244.
[28] J. Hayet, C. Esteves, and R. Murrieta-Cid, "A motion planner for maintaining landmark visibility with a differential drive robot," in *Algorithmic Foundations of Robotics VIII*. Berlin, Germany: Springer, 2009.
[29] P. Salaris, D. Fontanelli, L. Pallottino, and A. Bicchi, "Shortest paths for a robot with nonholonomic and field-of-view constraints," *IEEE Trans. Robot.*, vol. 26, no. 2, pp. 269–281, Apr. 2010.
[30] S. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.
[31] M. Baumann, S. Leonard, E. Croft, and J. Little, "Path planning for improved visibility using a probabilistic road map," *IEEE Trans. Robot.*, vol. 26, no. 1, pp. 195–200, Feb. 2010.
[32] Z. Yao and K. Gupta, "Path planning with general end-effector constraints," *Robot. Auton. Syst.*, vol. 55, no. 4, pp. 316–327, 2007.
[33] M. Kazemi, K. Gupta, and M. Mehrandezh, "Global path planning for robust visual servoing in complex environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 326–332.
[34] G. Morel, P. Zanne, and F. Plestan, "Robust visual servoing: Bounding the task function tracking errors," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 6, pp. 998–1009, Nov. 2005.
[35] F. Lamiraux, E. Ferre, and E. Vallee, "Kinodynamic motion planning: Connecting exploration trees using trajectory optimization methods," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr.–May 2004, vol. 4, pp. 3987–3992.
[36] I. Belousov, C. Esteves, J.-P. Laumond, and E. Ferre, "Motion planning for the large space manipulators with complicated dynamics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Aug. 2005, pp. 2160–2166.
[37] P. Corke and M. Good, "Dynamic effects in visual closed-loop systems," *IEEE Trans. Robot.*, vol. 12, no. 5, pp. 671–683, Oct. 1996.
[38] D. Dementhon and L. Davis, "Model-based object pose in 25 lines of code," *Int. J. Comput. Vis.*, vol. 15, nos. 1–2, pp. 123–141, 1995.
[39] E. Marchand and F. Chaumette, "Virtual visual servoing: A framework for real-time augmented reality," in *Proc. Eurograph. Conf.*, Saarbrucken, Germany, 2002, vol. 21, pp. 289–297.
[40] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. Autom*, 2000, vol. 2, pp. 995–1001.
[41] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, 2001.
[42] D. Baraff. (1997). An introduction to physically based modeling: Rigid body simulation I—Unconstrained rigid body dynamics, in *Proc. SIGGRAPH Course Notes* [Online]. Available: http://www.cs.cmu.edu/baraff/pbm/pbm.html
[43] D. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Mach. Syst.*, vol. MMS-10, no. 2, pp. 47–53, Jun. 1969.
[44] J. J. Craig, *Introduction to Robotics: Mechanics and Control*. Reading, MA, USA: Addison-Wesley, 1989.
[45] I. Gipson, K. Gupta, and M. Greenspan, "MPK: An open extensible motion planning kernel," *J. Robot. Syst.*, vol. 18, no. 8, pp. 433–443, 2001.
[46] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: A generic software platform with a wide class of robot control skills," *IEEE Robot. Autom. Mag.*, vol. 12, no. 4, pp. 40–52, Dec. 2005.
[47] R. Diankov and J. Kuffner. (2008, Jul.). OpenRAVE: A planning architecture for autonomous robotics. Robotics Institute, Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-08-34 [Online]. Available: http://openrave.programmingvision.com
[48] F. Chaumette and S. Hutchinson, "Visual servo control Part I: Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, no. 4, pp. 82–90, Dec. 2006.
[49] E. Malis and P. Rives, "Robustness of image-based visual servoing with respect to depth distribution errors," in *Proc. IEEE Int. Conf. Robot. Autom.*, Sep. 2003, vol. 1, pp. 1056–1061.
[50] B. Tamadazte, N. Piat, and S. Dembele, "Robust trajectory tracking and visual servoing schemes for MEMS manipulation," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron.*, Jul. 2010, pp. 860–865.
[51] M. Kazemi, K. Gupta, and M. Mehrandezh, "Path planning for image-based control of wheeled mobile manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2011, pp. 5306–5312.

Moslem Kazemi received the Bachelor's degree in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2000, the Master's degree in industrial systems engineering from the University of Regina, SK, Canada, in 2004, and the Ph.D. degree in engineering science from Simon Fraser University, Burnaby, BC, Canada, in 2012.

He has been involved in various robotics projects from mobile robot control and navigation to force compliant manipulation and grasping. He was a member of SharifCE RoboCup middle-size robot soccer players team from 1997 to 2002 and won the RoboCup World Championships in 1999 and 2000, and the Europe Championships in 2000. He is in close collaboration with the Geography Department of Simon Fraser University, where he leads the software design and integration of a smart data acquisition cart for a hydraulic flume. In 2011, he joined the Robotics Institute, Carnegie Mellon University (CMU), Pittsburgh, PA, USA, as a Postdoctoral Fellow, where he was involved in the development of force compliant grasping and manipulation strategies for Defense Advanced Research Projects Agency Autonomous Robotic Manipulation-Software Track (ARM-S) challenge and is currently a Project Scientist leading the CMU ARM-S manipulation software developments. His main research interests include robotic manipulation and grasping, path planning and vision-based control of robotic arms, and smart systems software/hardware integration.

Kamal K. Gupta (M'87) received the B.Tech. degree from the Indian Institute of Technology Delhi, New Delhi, India, in 1978, and the M.Eng. and Ph.D. degrees from McGill University, Montreal, QC, Canada, in 1981 and 1987, respectively, all in electrical engineering.

Since 1987, he has been a faculty member with the School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada, where he is currently a Professor and the Director of the school. He has held Visiting Scientist positions with the Institut National de Recherche en Informatique et Automatique, Rhône-Alpes, France, and with the Robotics Lab, Stanford University, Stanford, CA, USA. His research interests include the geometric aspects of robotics and automation, in particular motion planning, manipulation, and geometric reasoning, and 3-D vision for robotic tasks. His recent research is focused on integrating sensing with path planning and tackling the resulting issues for general robot-sensor systems.

Dr. Gupta was the Co-Chair of the Technical Committee on Motion and Path Planning, IEEE Robotics and Automation Society, from 1996 to 2004.

Mehran Mehrandezh (M'93) received the B.Sc. degree from the Sharif University of Technology (formerly known as Arya Mehr), Tehran, Iran, in 1990, the M.Sc. degree from Queen's University, Kingston, ON, Canada, in 1995, and the Ph.D. degree from the University of Toronto, Canada, in 1999, all in mechanical engineering.

From 1999 to 2001, he was a Postdoctoral Researcher with Simon Fraser University, Vancouver, BC, Canada. In 2002, he joined the Faculty of Engineering and Applied Science, University of Regina, SK, Canada, in 2002, where he is currently an Associate Professor. He holds an Adjunct Faculty position with the University of British Columbia, Vancouver, Canada, as well as an Associate position with the School of Kinesiology and Health Studies, University of Regina. His expertise is in mechatronics, control systems, unmanned vehicle systems, instrumentation, sensor fusion, and machine vision. He has published more than 60 articles in refereed journals and conference proceedings. He has supervised one Ph.D. and 12 M.Sc. students to the completion of their theses. The work done in his group on design and development of a pipe inspecting machine has been highlighted in *popular mechanics magazine* as one of the five high-tech fixes to infrastructure. This invention has also been broadcasted with the Canadian Broadcasting Corporation and with the Discovery Channel Canada. He has been awarded research grants by the Natural Sciences and Engineering Research Council, Canada, under the following categories: Strategic, Idea-to-Innovation, Industry-ENGAGE, and the Discovery grant programs.

Dr. Mehrandezh has been a member of the American Society of Mechanical Engineers since 1993. He is currently the Vice President of the IEEE South Saskatchewan Section.