2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

Collision-Free Visual Servoing of an Eye-in-Hand Manipulator via Constraint-Aware Planning and Control

Ambrose Chan, Simon Leonard, Elizabeth A. Croft, James J. Little

***Abstract*— A *constraint-aware* eye-in-hand visual servoing control law is proposed. The control law is designed for a robot manipulator with an uncalibrated camera mounted on its end-effector. The method uses an image-based command to describe the desired end-effector position with respect to an object with an unknown position. When starting from an unknown position, the control law uses feedback from the camera to move the robot towards the reference image while satisfying a set of system constraints. The visual servoing control law is implemented via a nonlinear model predictive control framework to generate feasible and realistic robot trajectories that respect the robot's joint limits and velocity limits. The control law explicitly keeps the target object within the camera's field of view and avoids potential collisions with workspace obstacles. An appropriate representation of the robot's whole-arm collision constraints is extracted from well-known path planning methods, such as probabilistic road maps and dynamic collision checking algorithms. Experiments using an uncalibrated eye-in-hand platform demonstrate the ability of the visual servoing control law to achieve closed-loop positioning via collision-free trajectories, even when the initial object location is uncertain.**

I. Introduction

Visual servo control of robotic manipulators has been an active area of research for the past 30 years [1][2][3][4]. While most image-based visual servoing (IBVS) methods perform well for correcting small errors, they are inherently unaware of the sensing, control, and physical limits of the robotic-camera system. Application of typical IBVS to large robot motions can result in unstable trajectories, due to: (i) the target object leaving the sensor's field-of-view; (ii) the robot being commanded to move beyond its mechanical limits, or beyond its actuation limits; or (iii) the robot arm physically colliding into other objects within its workspace. This work aims to simultaneously address all three of the above challenges for visual servoing of large robot motions.

In previous work, control schemes using hybrid 2-1/2D features [5], shifting cylindrical coordinates [6], image moments [7], and partitioned z-axis control [8] have been used by researchers to improve the Cartesian trajectory of the camera. These approaches, under specific conditions, lead to improved robot motions. To keep the target object within the camera's field of view during position-based visual servoing (PBVS), researchers have used an adaptive switching controller [9], 3D features [10], and alternative coordinate frames [11]. In [12], a secondary task function was used to avoid unilateral constraints by exploiting manipulator redundancy via gradient projection methods. Image-based path planning via homography-based reconstruction and artificial potential fields was proposed by [13] to manage field-of-view limits and joint limits. Circular-like trajectories were used in [14] and offline optimization methods were used in [15] [16] to realize large camera motions. More recently, model predictive control is used to control a robotic tool via feedback from a stationary imaging sensor [17] [18] and from a catadioptric camera in simulation [19].

The main contributions of this paper are two-fold: (i) the formulation of Model Predictive Control (MPC) for *eye-in-hand* visual servoing and, to our knowledge, its first experimental validation on a test-bed; and (ii) the inclusion of manipulator collision avoidance in MPC visual servoing using a novel representation of whole-arm collision constraints. We show that the collision-space manifold can be represented as a bounded approximation that not only is suitable for MPC optimization, but is also freely available from results of well-known path planning methods, such as probabilistic road maps (PRM) and dynamic collision checking (DCC).

MPC uses a dynamic model of the plant for online planning and selection of an optimal sequence of control actions to achieve a desired output, while compensating for prediction errors using feedback provided by observations of the real plant. MPC is formulated as the online solution of a finite horizon, open-loop, optimal control problem. This optimization is subject to the system dynamics and the constraints related to the states and inputs of the system. Details on the theory and analysis of MPC can be found in [20].

II. Eye-in-Hand Visual Servoing

A. System Modelling

The following shorthand notation is used to refer to the coordinate frames used for system modelling: robot base frame $\mathcal{F}_b$; robot end-effector frame $\mathcal{F}_e$; camera frame $\mathcal{F}_c$; and target object frame $\mathcal{F}_o$.

1) Object Model: The object model consists of a set of feature points, whose coordinates $({}^oX_j, {}^oY_j, {}^oZ_j)$ are defined with respect to $\mathcal{F}_o$. For a target object made up of n feature points, the object model used for prediction is ${}^o\mathbf{P}_j = \begin{bmatrix} {}^oX_j & {}^oY_j & {}^oZ_j & 1 \end{bmatrix}^T, j \in [1, n]$.

This work was supported in part by the Natural Science and Engineering Research Council of Canada, the Canada Foundation for Innovation and the UBC Institute for Computing, Information and Cognitive Systems.

Ambrose Chan is with the Dept. of Mechanical Engineering, University of British Columbia, Vancouver, BC, Canada (achan82@mech.ubc.ca)
Simon Leonard is with the Dept. of Computer Science, Johns Hopkins University, Baltimore, Maryland, United States (sleonard@jhu.edu)
Elizabeth A. Croft is with the Dept. of Mechanical Engineering, University of British Columbia, Vancouver, BC, Canada (ecroft@mech.ubc.ca)
James J. Little is with the Dept. of Computer Science, University of British Columbia, Vancouver, BC, Canada (little@cs.ubc.ca)

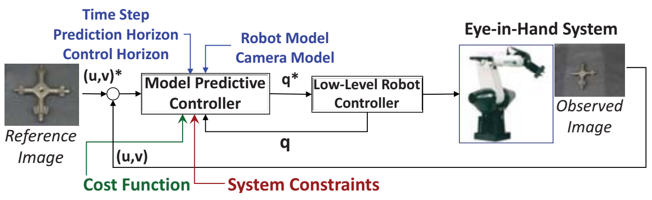


Fig. 1. A system diagram illustrating the input and output requirements of the MPC controller.

Let ${}^b\mathbf{T}_o$ define the homogeneous transformation that expresses the coordinates of the object frame in the coordinates of the robot base frame. The approximate object pose ${}^b\mathbf{T}_{\hat{o}}$ that is required for online open-loop planning in MPC is obtained from: (i) the camera, via pose estimation only using uncalibrated parameters; or (ii) the user, in the form of a rough pose of the target object, when initiating the robot positioning task.

2) Robot Model: A kinematic model of the robot ${}^b\mathbf{T}_e = f_{\text{robot}}(\mathbf{q})$, expresses the coordinates of the end-effector frame in the coordinates of the robot base frame. Kinematic models are typically used in visual servoing, given the relatively slow frame-rate of the camera with respect to the inner control loop of the robot, which stabilizes the dynamics. The objective of the MPC controller is to generate kinematically *feasible* robot trajectories that drive the robot towards completion of the positioning task using visual feedback. The joint trajectory, $\mathbf{q}^*(t)$, generated by MPC is then used as a reference signal (with quintic-polynomial interpolation) to be tracked by a low-level PID controller. A system diagram is shown in Figure 1.

3) Camera Model: A pin-hole camera model is used to describe the projection of the target object on the camera CCD array. Let $\mathbf{C}$ be the camera matrix defined as:

$$\mathbf{C} = \begin{bmatrix} fk_u & fk_u \cot\beta & u_0 \\ 0 & fk_v(\frac{1}{\sin\beta}) & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

where (u_0, v_0) are the image coordinates of the principal point, f is the focal length, β is the perpendicular skew angle, and k_u and k_v are the number of pixels per unit distance in x and y, respectively. The camera is mounted on the robot's distal link in an *eye-in-hand* configuration, such that the position and orientation of its canonical frame is controlled by the robot. The homogeneous transformation, ${}^c\mathbf{T}_e$, expresses the coordinates of the end-effector frame in the coordinates of the camera frame. The image coordinates (u_j, v_j) of the feature points can be modelled as:

$$(\mathbf{p}_j) = \begin{pmatrix} u_j \\ v_j \\ 1 \end{pmatrix} = \mathbf{C}[\mathbf{I}_{3\times3}|\mathbf{0}_{3\times1}]{}^c\mathbf{T}_e{}^e\mathbf{T}_b{}^b\mathbf{T}_o{}^o\mathbf{P}_j, \quad j \in [1, n]. \tag{2}$$

Finally, a vector consisting of the image coordinates of all feature points is used for image-based feedback control:

$$\overline{\mathbf{p}} = \begin{bmatrix} (\mathbf{p}_1)^T & (\mathbf{p}_2)^T & \cdots & (\mathbf{p}_n)^T \end{bmatrix}^T. \tag{3}$$

B. Constraint Modelling

1) Robot Joint-Limit and Joint-Velocity Constraints: For a robot with N independent joint actuators, the robot joint-limit constraints are modelled as lower bounds and upper bounds on the range of feasible joint positions and of feasible joint velocities:

$$\mathbf{q} \in [\mathbf{q}^{min}, \mathbf{q}^{max}], \qquad \mathbf{q}^{min}, \mathbf{q}^{max} \in \mathbb{R}^N, \tag{4}$$
$$\dot{\mathbf{q}} \in [\dot{\mathbf{q}}^{min}, \dot{\mathbf{q}}^{max}], \qquad \dot{\mathbf{q}}^{min}, \dot{\mathbf{q}}^{max} \in \mathbb{R}^N. \tag{5}$$

2) Camera Field-of-View Constraints: A rectangular imaging sensor array, whose maximum and minimum coordinates are u^{min} and u^{max} (respectively in the horizontal axis) and v^{min} and v^{max} (respectively in the vertical axis), has the following field-of-view constraints while viewing a target object with n features:

$$u_j(\mathbf{p}_j) \in [u^{min}, u^{max}] \qquad \forall j \in [1, n], \tag{6}$$
$$v_j(\mathbf{p}_j) \in [v^{min}, v^{max}] \qquad \forall j \in [1, n]. \tag{7}$$

3) Whole-Arm Collision Constraints: For a robot with N independent joint actuators, let $\mathbb{Q}_{free} \in \mathbb{R}^N$ be the collision-free space, consisting of the set of joint configurations that *do not* cause the robot to occupy the same physical space as a workspace object. Whole-arm collision constraints are represented as:

$$\mathbf{q} \in \mathbb{Q}_{free}. \tag{8}$$

An exact closed-form solution to $\mathbb{Q}_{free}$ is typically not available, except in the case of trivial workspace objects (such as points) with kinematically simplified robots (such as Cartesian gantry robots, or robots where N is limited). The discussion of a representation of collision-free space that is useful for MPC visual servoing is deferred until Section IV.

C. Control Law Design

A first-order discretization is used for MPC prediction for the visual feedback loop, where δ_t is the time step:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \delta_t \dot{\mathbf{q}}_k. \tag{9}$$

The error function for MPC visual servoing is chosen to be similar to IBVS. Let $\boldsymbol{\epsilon}_{i|k}$ represent the error predicted at time k for time $k+i$. This error is defined as the difference between the predicted image coordinates $\overline{\mathbf{p}}_{k+i|k}$ and the desired image coordinates $\overline{\mathbf{p}}^d$, plus a disturbance (model) correction term $\mathbf{d}_k$. The correction term $\mathbf{d}_k$ captures the difference between the latest plant output $\overline{\mathbf{p}}_k$ and the latest model output $\overline{\mathbf{p}}(\mathbf{q}_{k|k-1})$:

$$\boldsymbol{\epsilon}_{i|k} = (\overline{\mathbf{p}}_{k+i|k} - \overline{\mathbf{p}}^d) + \mathbf{d}_k, \tag{10}$$
$$\mathbf{d}_k = \overline{\mathbf{p}}(\mathbf{q}_{k|k-1}) - \overline{\mathbf{p}}_k. \tag{11}$$

The cost function $\mathcal{C}$ to be minimized is a quadratic function [18] of image errors $\boldsymbol{\epsilon}$ and of joint velocities $\dot{\mathbf{q}}$:

$$\mathcal{C} = \frac{1}{2}\left(\sum_{i=1}^{N_p-1} \boldsymbol{\epsilon}_{k+i|k}^T \mathbf{Q} \boldsymbol{\epsilon}_{k+i|k} + \dot{\mathbf{q}}_{k+i|k}^T \mathbf{W} \dot{\mathbf{q}}_{k+i|k}\right), \tag{12}$$

where N_p is the prediction horizon, and $\mathbf{Q}$ and $\mathbf{W}$ are two symmetric positive definite matrices, which weight the

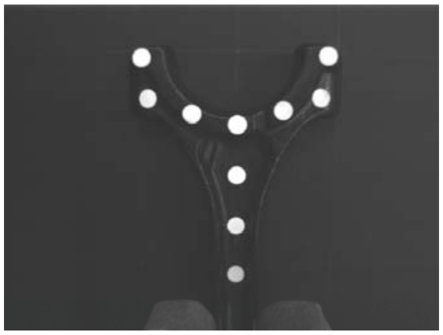


Fig. 2. Reference image describing the desired position of the end-effector with respect to a target object.

predicted image errors against the predicted joint control efforts at time instant k.

III. Simulations

A. Setup

The MPC eye-in-hand visual servoing controller is implemented in simulation to test its ability to perform large robot motions in the presence of constraints and pose estimation errors. The simulations are designed to reflect the parameters of the experimental test-bed described in Section V-A. The target object consists of ten non-coplanar feature points, resembling the object shown in Figure 2. In Figure 3, the feature points designated 'o' show the camera's initial view of the target object, while the feature points designated 'x' show the desired view. In the simulations, the *actual* pose of the target object is ${}^{b}\mathbf{T}_{o} = \mathbf{T}_{xyz}(-0.28, 0.51, 0.14)\mathbf{T}_{Rz}(145°)\mathbf{T}_{Rx}(-55°)[m]$, whereas the *estimated* pose of the target object used by the MPC controller for joint-space planning, is ${}^{b}\mathbf{T}_{\hat{o}} = \mathbf{T}_{xyz}(-0.30, 0.50, 0.15)\mathbf{T}_{Rz}(135°)\mathbf{T}_{Rx}(-45°)[m]$.

The prediction horizon N_p is set to 5 and the control horizon N_c is set to 1. The time step δ_t of 10ms is used for simulations. $\mathbf{Q}$ is the identity matrix, $\mathbf{I}$, while $\mathbf{W}$ is $2500\mathbf{I}$. The minimization of the cost function for predictive control is solved using a sequential quadratic programming (SQP) optimization algorithm [20][21]. To speed up computations, the gradients to the cost function and constraints are provided via a closed-form analytic function composed of the robot Jacobian, the image Jacobian and the measured errors.

B. Results

Figure 3 shows the target object's trajectory as observed by the eye-in-hand camera, while Figure 4 shows the camera's trajectory in Cartesian space. The motion requires significant translation towards the target object, as well as in-plane and out-of-plane rotations. The control law successfully generates motions that bring the camera to the desired position.

Figure 5 shows the joint-space trajectory of the 6-DoF CRS-A465 robot. The joint limits of the robot are shown as dotted lines, whereas the actual joint trajectories are shown as solid lines. The robot responds to its joint limits (e.g., joint 5) by predicting several steps ahead and replanning

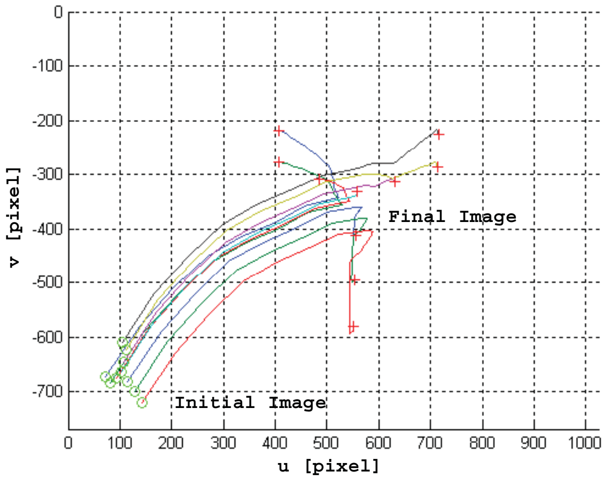


Fig. 3. Image trajectory generated by MPC visual servoing

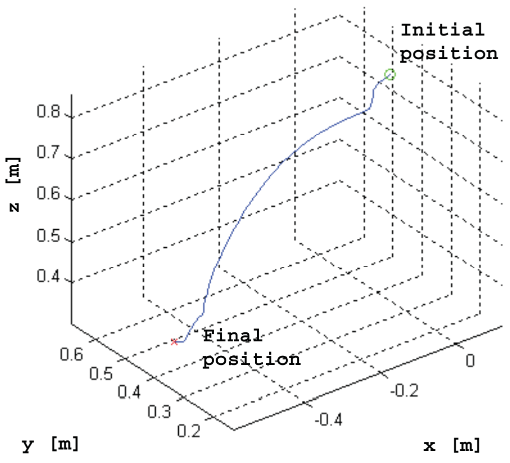


Fig. 4. Camera Cartesian trajectory generated by MPC visual servoing.

control inputs to avoid constraint violation. Figure 6 shows the joint velocities of the robot. The velocity limits of the robot are shown as dotted lines, whereas the actual joint velocities are shown as solid lines. The aggressive controller maximizes the robot's output capabilities by keeping the joint velocities just below their limits. Note that the simultaneous avoidance of these unilateral constraints cannot be achieved with a saturator, as this will likely bring the target object outside of the camera's field of view and cause instability in the control loop.

IV. Whole-Arm Collision Constraints

The representation of whole-arm collision constraints in terms of the robot's joint-space from known Cartesian-space geometry is a non-trivial problem. Figure 7 depicts a hypothetical set of collision-free robot configurations, represented as a three-dimensional manifold in joint-space for 3 out of the 6 available robot joints. While it is easy to check for the absence or presence of collision at a *particular* robot configuration, it is difficult to obtain topological information

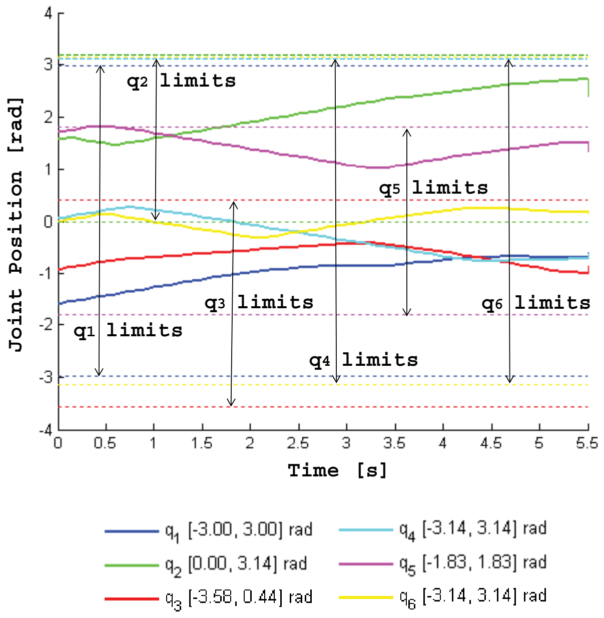


Fig. 5. Robot joint trajectory generated by MPC visual servoing, demonstrating the avoidance of robot joint position limits.

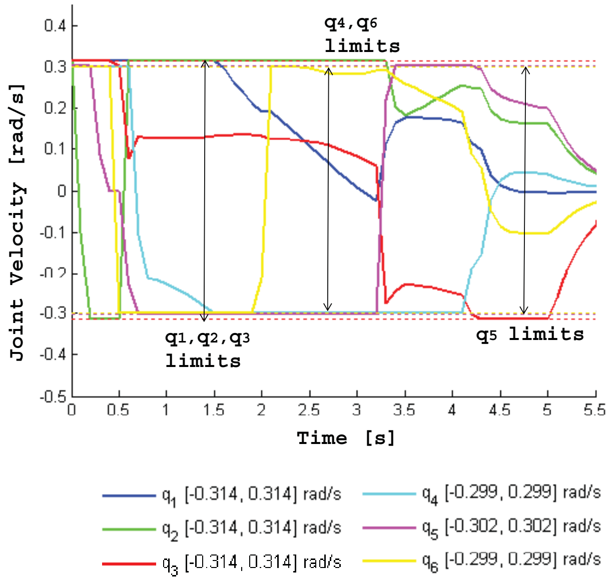
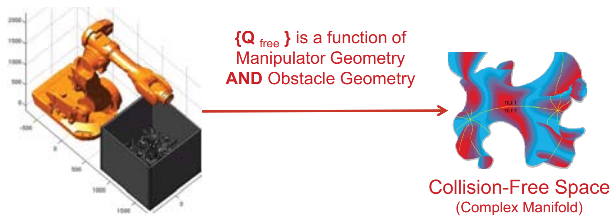


Fig. 6. Robot joint velocities generated by MPC visual servoing, demonstrating the avoidance of robot joint velocity limits.



{Q free } is a function of Manipulator Geometry AND Obstacle Geometry
Collision-Free Space
(Complex Manifold)

Fig. 7. Joint-space representation of the collision-free space (right) for a particular manipulator and obstacle configuration (left). The collision-free space is represented in 3-dimensional space for 3 joints only.

pertaining to the collision-free manifold, without which the constrained optimization problem cannot be easily solved.

Popular path-planning methods, such as probabilistic road maps (PRM), use sampling to first find collision-free configurations and then employ geometric collision-tests to determine if a *path* between two configurations is also collision-free. Unfortunately, such offline graph-based representations of the collision-free space are insufficient for online control, as one cannot guarantee that the robot will remain on the path while servoing. This section describes how an existing PRM graph or a successful collision-test between two robot configurations can be exploited to provide a representation of the collision-free space for MPC visual servoing.

A. Dynamic Collision Checking

Given the initial configuration $\mathbf{q}_i$ and the final configuration $\mathbf{q}_f$, a dynamic collision checking (DCC) algorithm proposed by [22] determines if the trajectory defined by $\mathbf{q}(t) = \mathbf{q}_i + (\mathbf{q}_f - \mathbf{q}_i)t$ for $0 \le t \le 1$ results in a collision. Let $d(\mathbf{q}_i)$ be the shortest distance between an obstacle and the robot in the configuration $\mathbf{q}_i$. Similarly, let $d(\mathbf{q}_f)$ be the shortest distance between an obstacle and the robot in the configuration q_f. Let $\ell(\mathbf{q}(t))$ be the the longest distance traveled by any point on the robot during the trajectory $\mathbf{q}(t)$. Then, given $d(\mathbf{q}_i)$, $d(\mathbf{q}_f)$ and $\ell(\mathbf{q}(t))$, DCC determines that the manipulator does not collide with the obstacle if:

$$\ell(\mathbf{q}(t)) < d(\mathbf{q}_i) + d(\mathbf{q}_f). \tag{13}$$

If (13) is not satisfied, the algorithm proceeds by dividing the trajectory $\mathbf{q}(t)$ in two trajectories $\mathbf{q}(u)$ for $0 \le u < {}^{t}\!/_{2}$ and $\mathbf{q}(v)$ for ${}^{t}\!/_{2} \le v \le 1$ and (13) is evaluated for both trajectories. This procedure is applied recursively until every trajectory satisfies (13) or a collision is detected.

Given a trajectory $\mathbf{q}(t)$ and the set of 3-D points $\mathbb{H}$ representing the hull of the manipulator, $\ell(\mathbf{q}(t))$ is determined by

$$\ell(\mathbf{q}(t)) = \max_{\mathbf{P}\in\mathbb{H}} \int_0^1 |\dot{\mathbf{P}}|_2 dt. \tag{14}$$

Solving (14) is tedious but much can be gained given that (13) is an inequality and it still holds if $\ell(\mathbf{q}(t))$ is replaced by an upper bound [22]. For a manipulator with N actuators, suppose that the trajectory $\mathbf{q}(t)$ only involves rotating the j^{th} actuator by an amount Δq_j. Then, it is possible to find a configuration for the actuators $q_{j+1}, \ldots, q_N$ such that

$$\ell(\mathbf{q}(t)) \le \max_{q_{j+1},\ldots,q_N} \ell(\mathbf{q}(t)) \tag{15}$$
$$= \ell_j(\mathbf{q}(t)). \tag{16}$$

If the j^{th} actuator is a revolute joint around the axis $\mathbf{z}_j$, then

$$\ell_j(\mathbf{q}(t)) = \max_{\substack{q_{j+1},\ldots,q_N \\ \mathbf{P}\in\mathbb{H}_j\cup\cdots\cup\mathbb{H}_N}} \int_0^1 |\dot{\mathbf{P}}|_2 dt \tag{17}$$
$$= r_j|\Delta q_j| \tag{18}$$

where $\mathbf{P} \in \mathbb{H}_j \cup \cdots \cup \mathbb{H}_N$ are the points on the hulls of the links j to N and r_j is the greatest possible distance

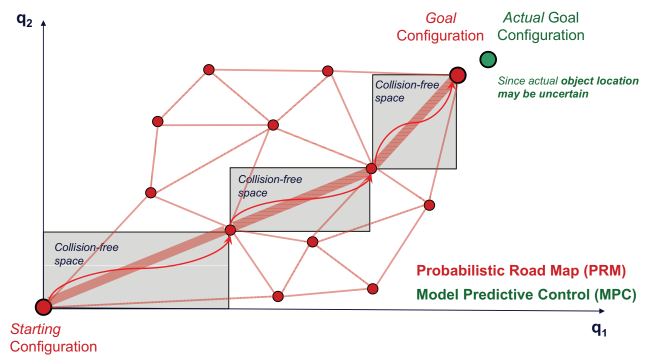

q2
Goal Configuration
Actual Goal Configuration
Since actual object location may be uncertain
Collision-free space
Collision-free space
Collision-free space
Probabilistic Road Map (PRM)
Model Predictive Control (MPC)
Starting Configuration
q1

Fig. 8. Hyperrectangular collision-free spaces obtained from a PRM as a byproduct of the form chosen for $\ell(\mathbf{q}(t))$ (2-DoF case shown here).

between the line $\mathbf{z}_j$ and a point $\mathbf{P}$. Finally, generalizing (17) to trajectories involving N actuators we obtain

$$\ell(\mathbf{q}(t)) \leq \sum_{j=1}^{N} r_j |\Delta q_j|. \tag{19}$$

B. Collision Constraints for Visual Servoing

The right hand side of (19) is "trajectory-free". That is, it does not depend on $\mathbf{q}_i$, $\mathbf{q}_f$ or the trajectory $\mathbf{q}(t)$. One notable constraint is that any trajectory must be bounded within the hyperrectangle with corners $\mathbf{q}_i$ and $\mathbf{q}_f$. Subsituting (19) into (13) results in:

$$\sum_{j=1}^{N} r_j |\Delta q_j| < d(\mathbf{q}_i) + d(\mathbf{q}_f). \tag{20}$$

Equation (20) is used to define collision-free regions that are: (i) obtained as a computational by-product of DCC and are (ii) expressed in a compact form that is suitable for online MPC visual servoing, as shown in Figure 8. If an existing PRM is not available, the collision-free space must be estimated dynamically as shown in Figure 9. At every iteration k, the optimal sequence of inputs determined by MPC is checked for collision prior to execution. If DCC fails, then adaptive bisection is invoked to provide an estimate of $\mathbb{Q}_{free}$ for replanning, starting at the initial configuration $\mathbf{q}_i$. The next optimization step replaces (8) with the joint boundaries defined by (20). This redefinition of the collision-free space allows the MPC controller to react to upcoming obstacles several steps ahead in the predictions horizon to achieve robustness against local minima.

V. Experiments

A. Experiment Setup

An overview of the experimental test-bed is shown in Figure 11. The aim of the experiment is to demonstrate positioning of the eye-in-hand robot with respect to a conrod located at a cluttered and uncertain location in the workspace, such that the two-fingered gripper is in the correct position for grasping. Figure 10 shows the initial image that the robot observes prior to servoing. To communicate the correct grasping position, the reference image in Figure 2 is shown to the robot.

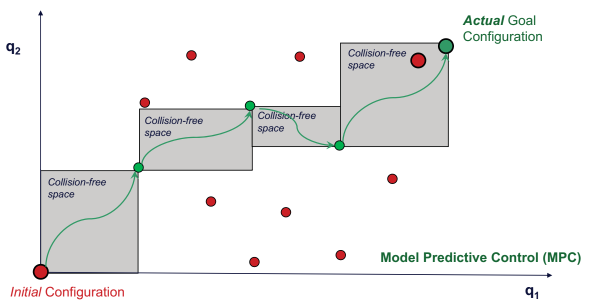


Fig. 9. Hyperrectangular collision-free spaces obtained from MPC and DCC as a byproduct of the form chosen for $\ell(\mathbf{q}(t))$ (2-DoF case shown here).

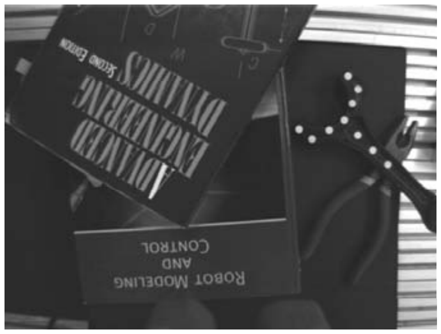


Fig. 10. Overhead image describing the initial position of the camera and end-effector.

1) Camera: The imaging sensor that is used in the experiments is a Sony XC-HR70 monochrome CCD camera, with a 4.8mm lens, as shown in Figure 11. The camera acquires images at a resolution of 1024×768 at a maximum frame rate of 30fps. Table I shows the camera parameters that are used in the MPC controller for experiments. These parameters are obtained from the product manual without calibration. A camera-to-end-effector transformation of ${}^{e}\mathbf{T}_c = \mathbf{T}_{Rz}(90°)\mathbf{T}_{Rx}(-12.5°)\mathbf{T}_{xyz}(0, -0.085, 0.070)[m]$ is measured by hand, roughly assuming the location of the optical center.

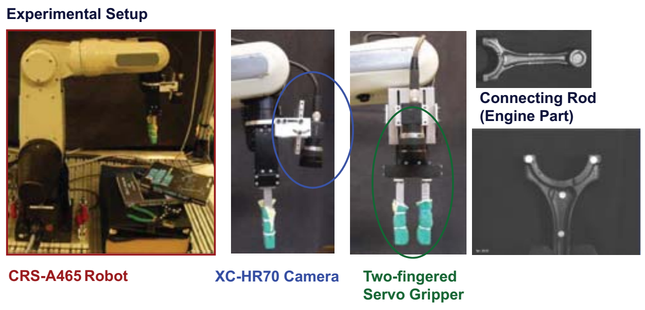


Fig. 11. Eye-in-hand platform for MPC visual servoing experiments.

TABLE I

UNCALIBRATED CAMERA PARAMETERS

f	0.0048	[m]
(u^{min}, u^{max})	(1,1023)	
(v^{min}, v^{max})	(1,767)	
(u_0, v_0)	(512,384)	
k_u	$(4.65 \times 10^{-6})^{-1}$	[pixels/m]
k_v	$(4.65 \times 10^{-6})^{-1}$	[pixels/m]
β	$90°$	

TABLE II

DIMENSIONS OF WORKSPACE OBSTACLES

	Length [m]	Width [m]	Height [m]
Textbook 1	0.255	0.175	0.025
Textbook 2	0.235	0.190	0.025
Binder	0.290	0.270	0.040
Pliers	0.165	0.075	0.020

2) *Robot:* The robot used in this experiment is a 6-DoF CRS-A465 robot. A forward kinematic model of the robot is used in the MPC controller to estimate for the location of the camera frame, given a set of joint configurations. Volumetric models of the robot, the camera, the gripper, and the obstacles (constructed using geometric primitives) are used for the dynamic collision tests. The joint-position limits and the joint-velocity limits (reduced to 5% of the allowed maximum for safety reasons) of the CRS-A465 robot are shown in Figure 5 and 6, respectively.

3) *Target Object:* The target object is an automobile part (conrod) with ten feature points as shown in Figures 10 and 2. In the experiments, a human user described the target object as having undergone a translation of $(10cm, 40cm, 15cm)$ in (x, y, z) and a rotation of $90°$ about the vertical axis, so that an object pose of ${}^{b}\mathbf{T}_{\hat{o}} = \mathbf{T}_{Rz}(90°)\mathbf{T}_{xyz}(0.100, 0.400, 0.150)[m]$ is used for joint-space path planning.

Positional errors have been introduced in order to demonstrate the robustness of MPC visual servoing with respect to modelling errors. Manual measurements confirm that the (x, y, z) location of the conrod with respect to the robot's base is closer to $(0.130, 0.460, 0.170)[m]$. One side of the conrod rests on top of another object, as shown in Figure 10, resulting in out-of-plane rotation in the y axis with respect to the robot's base. This is not captured in the pose estimate that is used by the MPC controller, which only has a rotation of $90°$ about the z axis. Also, the rotation of the conrod about the z axis is actually closer to $135°$. These additional pose estimation errors are designed to test the ability of the MPC controller to compensate for large errors in its prediction model.

4) *Workspace Obstacles:* The obstacles are modelled as polygons for collision detection. The dimensions of the workspace obstacles and the homogeneous transformations from their centroidal frames to the robot base frame can be found in Table II and Table III respectively.

5) *Control System Implementation:* The MPC controller is implemented on a Pentium Dual-Core 2.0 GHz CPU running Windows XP. A Matrox Meteor II acquisition board acquires images from the Sony XC-HR70 at a frame rate of 30Hz. A visual tracker based on the ViSP [23] library is used to track the centroid location of the feature points. The maximum control rate that is achieved, with SQP optimization running on one thread and feature tracking running on another, is 10Hz. The low-level PID, independent joint controller is implemented on a Pentium 4, 2.8 GHz computer operating with a Windows RTX extension. The hardware is controlled through a Quanser Multi-Q PCI card and WinCon software. The inner control loop runs at 1kHz. Quintic polynomial interpolation is used to provide smooth reference signals for PID tracking, to account for the difference in control rates between the inner and the outer control loops. Communication between controllers is handled via the Quanser serial block.

TABLE III

LOCATION OF WORKSPACE OBSTACLES

	Homogeneous Transformation [m]
Textbook 1	${}^{b}\mathbf{T}_{wo_1} = \mathbf{T}_{Rz}(60°)\mathbf{T}_{xyz}(-0.030, 0.570, 0.185)$
Textbook 2	${}^{b}\mathbf{T}_{wo_2} = \mathbf{T}_{Rz}(95°)\mathbf{T}_{xyz}(-0.025, 0.450, 0.160)$
Binder	${}^{b}\mathbf{T}_{wo_3} = \mathbf{T}_{xyz}(-0.040, 0.440, 0.120)$
Pliers	${}^{b}\mathbf{T}_{wo_4} = \mathbf{T}_{Rz}(60°)\mathbf{T}_{xyz}(0.110, 0.440, 0.150)$

6) *Tuning Parameters:* The prediction horizon N_p is set to 10 and the control horizon N_c is set to 1. The use of a longer prediction horizon in the MPC controller is intended to account for the (possibly significant) discrepancy between the real eye-in-hand system and the approximate model used by the MPC controller for planning. The MPC controller time step, δ_t, is set at 0.1 seconds. $\mathbf{Q}$ is an identity matrix, $\mathbf{I}$, while $\mathbf{W}$ is 10000$\mathbf{I}$. The difference in magnitude between the chosen $\mathbf{W}$ and the chosen $\mathbf{Q}$ accommodates unit normalization (i.e., pixels vs. joint angles in radians).

B. Experiment Results

Figure 12 shows the trajectory of the target object, observed from the eye-in-hand camera. The image-based control law successfully handles both the significant rotation that is required about the camera's optical axis and the translational motion that is required towards the target object. The MPC controller is able to recognize the out-of-plane rotation required to properly align the gripper with the conrod.

Figure 13 shows the trajectory of the robot as the visual servoing task is executed. The initial robot motions executed in frames 1 to 7 are very aggressive; the DCC algorithm is able to guarantee collision-free motion for all prediction steps ahead. Robot motions become more conservative when the robot gripper is in proximity to the workspace obstacles next to the conrod. In frames 8 to 12, collision-free motion can no longer be guaranteed by the DCC algorithm between large changes in robot configurations. As the regular DCC algorithm fails (inequality not satisfied), the MPC controller replaces its joint limits with the dynamically updated collision-free bounds provided by the DCC bisection algorithm. The

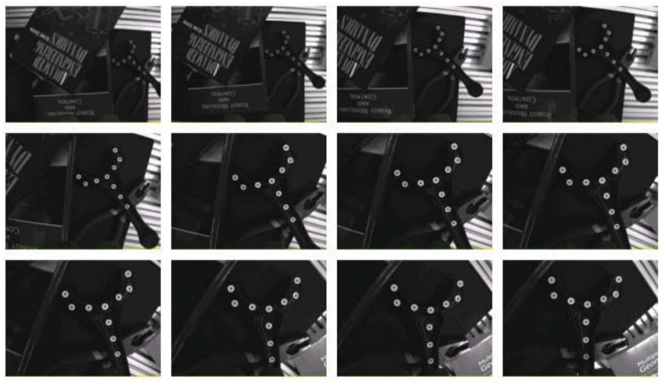


Fig. 12. Sequence of camera motions generated by MPC visual servoing to complete a grasping task. The resulting image feature trajectories satisfy the camera field-of-view constraints.

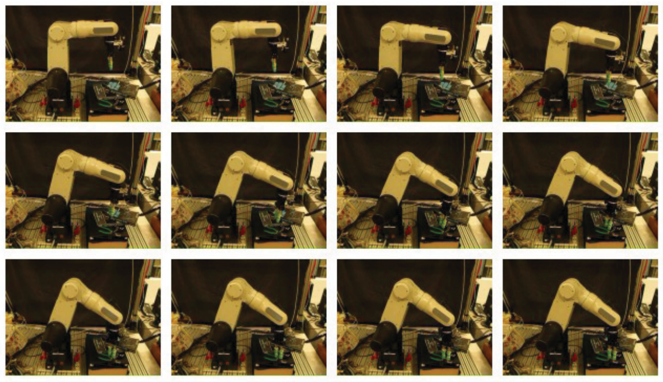


Fig. 13. Sequence of robot motions generated by MPC visual servoing to complete a grasping task. The resulting robot trajectory satisfies joint position, joint velocity, and workspace collision constraints.

MPC controller is optimized over a smaller region in joint-space where collision-free robot motion is feasible. The last few frames show the robot taking incremental steps within the latest, dynamically updated collision-free region while servoing towards the goal.

VI. Conclusion

Our constraint-aware control law for eye-in-hand manipulators successfully demonstrates visual servoing motions across the workspace while handling joint, visual and collision constraints. The MPC controller does not require the camera to be calibrated and only needs an approximate estimate of the target object location. Using the DCC algorithm, the control law can move quickly in uncluttered regions but still generates whole-arm, collision-free motions in cluttered areas. Future work will consider more complex scenarios as well as applications to robot arms on mobile platforms, increasing the workspace size.

References

[1] Seth Hutchinson, Gregory D. Hager, and Peter I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.

[2] Danica Kragic and Henrik I. Christensen. Survey on visual servoing for manipulation. Technical report, Computational Vision and Active Perception Laboratory, 2002.

[3] François Chaumette and Seth Hutchinson. Visual servo control part i: Basic approaches. *IEEE Robotics & Automation Magazine*, 13(4):82–90, December 2006.

[4] François Chaumette and Seth Hutchinson. Visual servo control part ii: Advanced approaches. *IEEE Robotics & Automation Magazine*, 14(1):109–118, March 2007.

[5] Ezio Malis and François Chaumette. Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods. *International Journal of Robotics and Automation*, 18(2):176–186, 2002.

[6] Masami Iwatsuki and Norimitsu Okiyama. A new formulation of visual servoing based on cylindrical coordinate system. *IEEE Transactions on Robotics*, 21(2):266–273, April 2005.

[7] François Chaumette. Image moments: a general and useful set of features for visual servoing. *IEEE Transactions on Robotics*, 20(4):713–723, August 2004.

[8] Peter I. Corke and Seth A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507–515, August 2001.

[9] Graziano Chesi, Koichi Hashimoto, Domenico Prattichizzo, and Antonio Vicino. A switching control law for keeping features in the field of view in eye-in-hand visual servoing. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, pages 3929–3934, Taipei, Taiwan, September 2003.

[10] E. Cervera, A. P. del Pobil, F. Berry, and P. Martinet. Improving image-based visual servoing with three-dimensional features. *The International Journal of Robotics Research*, 22(10-11):821–839, October-November 2003.

[11] Lingfeng Deng, Farrokh Janabi-Sharifi, and William J. Wilson. Hybrid motion control and planning strategies for visual servoing. *IEEE Transactions on Industrial Electronics*, 52(4):1024–1040, August 2005.

[12] Nicolas Mansard and François Chaumette. A new redundancy formalism for avoidance in visual servoing. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 468– 474, Edmonton, Canada, August 2005.

[13] Youcef Mezouar and François Chaumette. Path planning for robust image-based control. *IEEE Transactions on Robotics and Automation*, 18(4):534–549, August 2002.

[14] G. Chesi and A. Vicino. Visual servoing for large camera displacements. *IEEE Transactions on Robotics*, 20(4):724735, August 2004.

[15] G. Chesi and Y. S. Hung. Global path-planning for constrained and optimal visual servoing. *IEEE Transactions on Robotics*, 23(5):10501060, October 2007.

[16] G. Chesi. Visual servoing path-planning via homogeneous forms and lmi optimizations. *IEEE Transactions on Robotics*, 25(2):281291, April 2009.

[17] M. Sauvée, P. Poignet, E. Dombre, and E. Courtial. Image based visual servoing through nonlinear model predictive control. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 1776–1781, San Diego, CA, USA, December 2006.

[18] M. Sauvée, P. Poignet, and E. Dombre. Ultrasound image-based visual servoing of a surgical instrument through nonlinear model predictive control. *The International Journal of Robotics Research*, 27(1):25–40, January 2008.

[19] G. Allibert, E. Courtial, and Y. Tour. Visual predictive control for manipulators with catadioptric camera. In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, pages 510–515, Pasadena, USA, May 2008.

[20] J. M. Maciejowski. *Predictive Control with Constraints*. Pearson Education Limited, 2002.

[21] M.J.D. Powell. *Variable Metric Methods for Constrained Optimization*. Springer-Verlag, 1983.

[22] Fabian Schwarzer, Mitul Saha, and Jean-Claude Latombe. Adaptive dynamic collision checking for single and multiple articulated robots in complex environments. *IEEE Transactions on Robotics*, 21(3):338–353, June 2005.

[23] E. Marchand, F. Spindler, and F. Chaumette. Visp for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine*, 12(4):40–52, December 2005.