Online Camera Registration for Robot Manipulation

Conference Paper · June 2014

DOI: 10.1007/978-3-319-23778-7_13

Neil Dantam
Rice University

Heni Ben Amor
Arizona State University

Henrik Iskov Christensen
University of California, San Diego

Online Camera Registration for Robot Manipulation

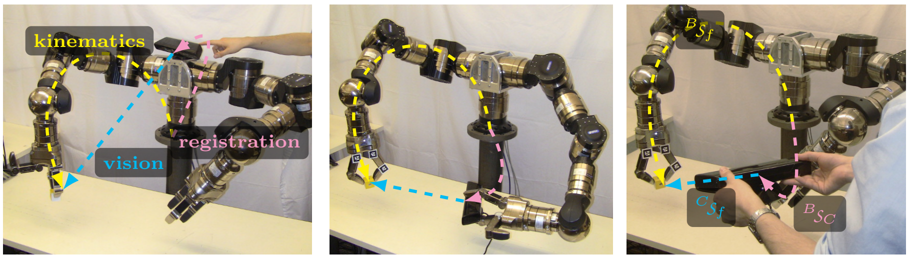
Neil Dantam, Heni Ben Amor, Henrik Christensen, and Mike Stilman*

Institute for Robotics and Intelligent Machines, Georgia Institute of Technology
Atlanta, GA 30332, USA.
ntd@gatech.edu, hbenamor@cc.gatech.edu,
hic@cc.gatech.edu, mstilman@cc.gatech.edu

Abstract. We demonstrate that millimeter-level manipulation accuracy can be achieved without the static camera registration typically required for visual servoing. We register the camera online, converging in seconds, by visually tracking features on the robot and filtering the result. This online registration handles cases such as perturbed camera positions, wear and tear on camera mounts, and even a camera held by a human. We implement the approach on a Schunk LWA4 manipulator and Logitech C920 camera, servoing to target and pre-grasp configurations. Our filtering software is available under a permissive license.[1]

1 Introduction

Using visual feedback for robot manipulation requires registration between the camera and the manipulator. Typically, this is viewed as a static task: registration is computed offline and assumed to be constant. In reality, camera registration changes during operation due to external perturbations, wear and tear,



Camera Perturbation Controlled Motion Uncontrolled Motion

Fig. 1. Use cases for online camera registration. We combine the visual and kinematic pose estimates of end effector and filter the result to estimate the camera pose in robot body frame.

*This work supported by a grant from Peugeot S.A.

[1] software available at http://github.com/golems/reflex

or even human repositioning. For example, during the recent DARPA Robotics Challenge trials, impacts from falls resulted in camera issues which significantly affected the robot behavior for some teams [10]. Fig. 1 shows additional use cases which may change the camera pose. The pose registration process should be treated as a dynamic task in which the involved parameters are continuously updated. Such an *online* approach to pose registration is challenging, since it requires the constant visibility of a calibration reference and sufficient accuracy to perform manipulation tasks.

To address changes in camera pose during operation, we propose an online camera registration method that combines (1) visual tracking of features on the manipulator, (2) a novel expectation-maximization inspired algorithm for pose filtering and tracking, and (3) an special Euclidean group constrained extended Kalman filter. Our key insight is to use the robot body as a reference for the registration process. By tracking known patterns or objects on the robot, we can continuously collect evidence for the current camera pose. However, naïve filtering of these pose estimates can lead to large variances in the calculated poses. The challenge is obtaining sufficient accuracy for manipulation through the online registration. To address this challenge, we combine pose filtering and manipulator control, incorporating camera registration into our manipulation feedback loop.

This paper presents a method for online registration and manipulation that combines object tracking, pose filtering, and visual servoing. First, we use perceptual information to identify the pose of specific features on the end-effector of the controlled robot (see Sect. 3.1). Then, we perform an initial fit to find offsets of the features on the robot, (see Sect. 3.2). A special Kalman filter is, then, used in conjunction with median filtering in order to perform online registration of the camera (Sect. 3.3). In our evaluation (see Sect. 4), we investigate the accuracy of the proposed method by applying it to robot grasping and manipulation tasks.

2 Related Work

Typical camera registration methods collect a set of calibration data using an external reference object, compute the calibration, then proceed assuming the calibration is static. OpenCV determines camera registration from point correspondes, typically using a chessboard [15]. Pradeep, et. al, develop a camera and arm calibration approach based on bundle adjustment and demonstrate it on the PR2 robot [16]. This approach requires approximately 20 minutes to collect data and another 20 minutes for computation, a challenge for handling changing pose online.

Visual servo control incorporates camera feedback into robot motion control [2,3]. The two main types of visual servoing are image-based visual servo control (IBVS), which operates on features in the 2D image, and position-based visual servo control, which operates on 3D parameters. Both of these methods assume a given camera registration. While IBVS is locally stable with regard to pose errors, under PBVS, even small pose errors can result in large tracking error [2]. Our proposed method addresses these challenges by correcting the camera registration online. In our experiments we show the importance of treating the registration process as a dynamic task. Furthermore, we show that our online registration achieves millimeter positioning accuracy of the manipulator. This is particularly important for grasping tasks performed using multi-fingered robot hands [1]. During such grasping tasks, inaccuracies in perception and forward kinematics often lead to premature contact between one finger and the object. As a result of the ensuing object movement, the intended grasp might not be satisfactorily executed or may fail altogether.

Other recent work has explored online visual parameter identification. [11] tracks a robot arm to identify encoder offsets. This method assumes a given camera registration, but is also tolerant of some registration error. In contrast, our work identifies the camera registration online, but does not explicitly consider encoder offsets. [19] uses maps generated from a Simultaneous Localization and Mapping (SLAM) algorithm to calibrate a depth sensor. In our approach, unlike typical environments for SLAM, the object to which we are trying to register our camera – the manipulator – will necessarily be in motion.

3 Technical Approach

We determine the pose registration between the camera and the manipulator by visually tracking the 3D pose of the arm. We identify the pose of texture or shape features on the arm and fit a transformation based on the corresponding kinematic pose estimates of those features. To obtain sufficient accuracy for manipulation, we combine several methods to fit and filter the visual pose esti-

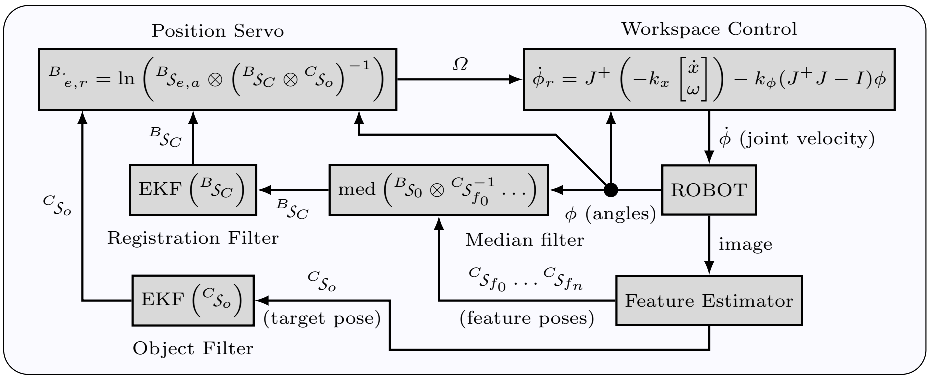

Position Servo
Workspace Control
${}^{B}\dot{}_{e,r} = \ln\left({}^{B}\mathcal{S}_{e,a} \otimes \left({}^{B}\mathcal{S}_{C} \otimes {}^{C}\mathcal{S}_{o}\right)^{-1}\right)$
Ω
$\dot{\phi}_r = J^{+}\left(-k_x \begin{bmatrix}\dot{x}\\ \omega\end{bmatrix}\right) - k_{\phi}(J^{+}J - I)\phi$
${}^{B}\mathcal{S}_{C}$
$\dot{\phi}$ (joint velocity)
EKF $\left({}^{B}\mathcal{S}_{C}\right)$
med $\left({}^{B}\mathcal{S}_{0} \otimes {}^{C}\mathcal{S}_{f_0}^{-1} \ldots\right)$
ROBOT
${}^{C}\mathcal{S}_{o}$
${}^{B}\mathcal{S}_{C}$
ϕ (angles)
Registration Filter
Median filter
image
${}^{C}\mathcal{S}_{o}$
${}^{C}\mathcal{S}_{f_0} \ldots {}^{C}\mathcal{S}_{f_n}$
EKF $\left({}^{C}\mathcal{S}_{o}\right)$
(target pose)
(feature poses)
Feature Estimator
Object Filter

Fig. 2. Block Diagram of Control System. 3D poses for features are estimated from visual data. The median camera transform is computed over all features and then Kalman filtered. With this registration, the robot servos in workspace to a target object location.

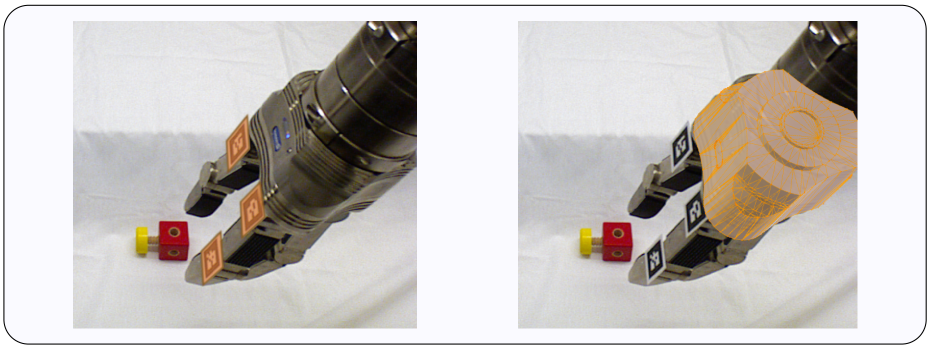


Fig. 3. Marker-based tracking (left) and model-based tracking (right).

mates before servoing to the target object. This estimation and control loop is summarized in Fig. 2.

For computational reasons, we used the *dual quaternion* representation for the special Euclidean group $\mathcal{SE}(3)$. Compared to matrices, the dual quaternion has lower dimensionality and is more easily normalized, both advantages for our filtering implementation. The relevant dual quaternion equations are summarized in appendix A. We represent the dual quaternion $\mathcal{S}$ for a transformation implicitly as a tuple of a rotation quaternion q and translation vector v: $\mathcal{S} = (q, v)$. This requires only seven elements. For Euclidean transformations, we use the typical coordinate notation where leading superscript denotes the parent frame and following subscript denotes the child frame, i.e., ${}^{x}\mathcal{S}_{y}$ gives the origin of y relative to x. The transformation ${}^{a}\mathcal{S}_{b}$ followed by ${}^{b}\mathcal{S}_{c}$ is given as the dual quaternion multiplication ${}^{a}\mathcal{S}_{b} \otimes {}^{b}\mathcal{S}_{c} = {}^{a}\mathcal{S}_{c}$.

3.1 Feature Estimation

To use the robot body as a reference for camera registration, it is important to identify and track body parts, e.g., the end-effector, in 3D. These 3D poses can be estimated with marker-based [17] and model-based approaches [4], see Fig. 3. Marker-based approaches require binary fiducials to known locations on the robot, such as the fingers. Model-based tracking, on the other hand, requires accurate polygon meshes of the tracked object. In our implementation, we use the ALVAR library [17] for marker-based tracking. For model-based tracking, we use the approach from [4]. In each frame the 3D pose of the object is computed by projecting a 3D CAD model into the 2D image. After projection, we identify salient edges in the model and align them with edges in the 2D image. A particle filter is then used to filter the pose estimates over time. Both marker-based and model-based tracking provide 3D pose estimates of tracked features, but with frequent outliers and noise. Markers have the advantage of being easy to deploy, while model-based tracking can deal with partial occlusions of the scene.

3.2 Offset Identification

To improve the accuracy of kinematic pose estimates for features, we initially perform a static expectation-maximization-like [6] procedure, based on the following model:

$$ {}^{B}\mathcal{S}_{k} \otimes {}^{k}\mathcal{S}_{f} = {}^{B}\mathcal{S}_{C} \otimes {}^{C}\mathcal{S}_{f} \tag{1} $$

where ${}^{B}\mathcal{S}_{k}$ is the measured nominal feature pose in the body frame determined from encoder positions and forward kinematics, ${}^{k}\mathcal{S}_{f}$ is the unknown static pose offset of the feature due to inaccuracy of manual placement, ${}^{B}\mathcal{S}_{C}$ is the unknown camera registration in the body frame, and ${}^{C}\mathcal{S}_{f}$ is the visually measured feature pose in the camera frame. These transforms are summarized in Fig. 1, with ${}^{B}\mathcal{S}_{k} \otimes {}^{k}\mathcal{S}_{f}$ combined as ${}^{B}\mathcal{S}_{f}$.

As an initialization step, we iteratively fix either ${}^{k}\mathcal{S}_{f}$ or ${}^{B}\mathcal{S}_{C}$ in (2) and solve for the other using Umeyama's algorithm [20]. This gives us the relative transforms for the features ${}^{k}\mathcal{S}_{f}$ which we assume are static.

3.3 Filtering

To compute the online registration, where ${}^{B}\mathcal{S}_{C}$ is changing, we combine median and Kalman filtering. The median filter is applied *independently* at each time step to reject major outliers in the estimated feature poses. Compared to weighted least squares methods, the median requires no parameter tuning and is especially resistant, tolerating outliers in up to 50% of the data [8]. Given the median at each step, the Kalman filter is applied *over time* to generate an optimal registration estimate under a Gaussian noise assumption.

Based on (2), each observed feature on the robot gives on estimate for the camera registration ${}^{B}\mathcal{S}_{C}$:

$$ {}^{B}\mathcal{S}_{k} \otimes {}^{k}\mathcal{S}_{f} \otimes ({}^{C}\mathcal{S}_{f})^{-1} = {}^{B}\mathcal{S}_{C} \tag{2} $$

Median Filtering At each time step, we find the median registration over all observed features. Each observed feature gives a candiate registration ${}^{B}\mathcal{S}_{C}$. First, we collect a set Q of the orientation candidates:

$$ Q = \left\{ ({}^{B}q_{C})_{i} \mid ({}^{B}\mathcal{S}_{k})_{i} \otimes {}^{k}\mathcal{S}_{f} \otimes ({}^{C}\mathcal{S}_{f})_{i}^{-1} \right\} \tag{3} $$

Then, we compute the median of the candidate orientation registrations Q. To find this median, the structure of rotations in $\mathcal{SO}(3)$offers a convenient distance metric between two orientations: the angle between them. Using this geometric interpretation, the median orientation $\hat{q}$ is the orientation with minimum angular distance to all other orientations.

$$ \widehat{{}^{B}q_{C}} = \arg\min_{q_i \in Q} \sum_{j=0}^{n} \left| \ln(q_i^{*} \otimes q_j) \right| \tag{4} $$

The median translation $\hat{x}$ is the conventional geometric median, the translation with minimum Euclidean distance to all other translations. First, we find the set of candidate translations Z by rotating the feature translation in camera frame ${}^{C}v_f$ and subtracting from the body frame translation ${}^{B}v_f$:

$$Z = \left\{ z_i \mid z_i = {}^{B}v_{f,i} - \widehat{{}^{B}q_C} \otimes {}^{C}v_{f,i} \otimes \widehat{{}^{B}q_C}^{*} \right\} \tag{5}$$

Then, we compute the geometric median of the candidate translations by finding the element with minimum distance to all other elements:

$$\widehat{{}^{B}v_C} = \underset{z_i \in Z}{\arg\min} \sum_{j=0}^{n} |z_i - z_j| \tag{6}$$

Then, the median transform is the combination of the orientation and translation parts:

$$\widehat{{}^{B}S_C} = \left(\widehat{{}^{B}q_C},\ \widehat{{}^{B}v_C} \right) \tag{7}$$

Kalman Filtering Next, we use an Extended Kalman filter (EKF) to attenuate noise over time, taking care to remain in the $\mathcal{SE}(3)$ manifold. Similar Kalman filters are discussed in [13,5]. The quasi-linearity of quaternions means the EKF is suitable for orientation estimation in this application [12].

To filter $\mathcal{SE}(3)$ poses, we consider state x composed of a quaternion q, a translation vector v, and the translational and rotational velocities, $\dot{v}$ and ω:

$$x = (q, v) = [q_x, q_y, q_z, q_w, v_x, v_y, v_z, \dot{v}_x, \dot{v}_y, \dot{v}_z, \omega_x, \omega_y, \omega_z]$$

The measurement z is the pose:

$$z = (q, v) = [q_x, q_y, q_z, q_w, v_x, v_y, v_z]$$

The general EKF prediction step for time k is:

$$\hat{x}_{k|k-1} = f(\hat{x_{k-1}}) \tag{8}$$

$$F_{k-1} = \left. \frac{\partial f}{\partial x} \right|_{\hat{x}_{k-1|k-1}} \tag{9}$$

$$P_{k|k-1} = F_{k-1} P_{k-1|k-1} F_{k-1}^{T} + Q_{k-1} \tag{10}$$

where $\hat{x}$ is the estimated state, $f(x)$ is the process model, F is the Jacobian of f, P is the state covariance matrix, and Q is the process noise model.

The process model then integrates the translational and rotational velocity, staying in the $\mathcal{SE}(3)$ manifold using the exponential of the twist Ω:

$$\Omega(\omega, \dot{v}, v) = (\omega, v \times \omega + \dot{v})$$

$$f(x) = \exp\left(\frac{\Delta t}{2} \Omega \right) \otimes (q, v) \tag{11}$$

Now, we find the process Jacobian F. The translation portion is a diagonal matrix of the translational velocity. For the orientation portion, we find the quaternion derivative $\dot{q}$ from the rotational velocity:

$$\dot{q} = \frac{1}{2}\omega \otimes q \tag{12}$$

This quaternion multiplication can be converted into the following matrix multiplication:

$$\begin{aligned} \frac{1}{2}\omega \otimes q &= \frac{1}{2}M_r(q)\,\omega \\ M_r(q) &= \begin{bmatrix} q_w & q_z & -q_y \\ -q_z & q_w & q_x \\ q_y & -q_x & q_w \\ -q_x & -q_y & -q_z \end{bmatrix} \end{aligned} \tag{13}$$

Note that we omit the w column of the typical quaternion multiplication matrix because the w element of rotational velocity ω is zero.

This gives the following process 13×13 Jacobian F:

$$F = \begin{bmatrix} I_{4\times4} & 0 & \frac{1}{2}\Delta t M_r\left(q\right) & 0 \\ 0 & I_{3\times3} & 0 & \Delta t I_{3\times3} \\ 0 & 0 & I_{3\times3} & 0 \\ 0 & 0 & 0 & I_{3\times3} \end{bmatrix} \tag{14}$$

Now we consider the EKF correction step. The general form is:

$$\hat{z}_k = h(\hat{x}_{k|k-1}) \tag{15}$$

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_{k|k-1}} \tag{16}$$

$$y_k = v(z_k, \hat{z}) \tag{17}$$

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{18}$$

$$H_k P_{k|k-1} = S_k K_k^T \tag{19}$$

$$\hat{x}_{k|k} = p(\hat{x}_{k|k-1}, K_k y_k) \tag{20}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{21}$$

where z is the measurement, h is the measurement model, H is the Jacobian of h, $\hat{z}$ is the estimated measurement, R is the measurement noise model, and K is the Kalman gain, v is a function to compute measurement residual, and p is a function to compute the state update.

We compute the EKF residuals and state updates using relative quaternions to remain in $\mathcal{SE}(3)$ without needing additional normalization. The observation $h(x)$ is a pose estimate:

$$\begin{aligned} h(x) &= (q, v) \\ H &= I_{7\times7} \end{aligned} \tag{22}$$

We compute the measurement residual based on the relative rotation between the measured and estimated pose:

$$\begin{aligned} v(z, \hat{z}) &= (y_q, y_v) \\ y_q &= \ln\left(z_q \otimes \hat{z}_q^*\right) \otimes q \\ y_v &= z_v - \hat{z}_v \end{aligned} \tag{23}$$

where y_q is the orientation part of the residual and y_v the translation part. Note that that $\ln\left(z_q \otimes \hat{z}_q^*\right)$ corresponds to a velocity in the direction of the relative transform between the actual and expected pose measurement and that we can consider y_q as a quaternion derivative. Then, the update function will integrate the pose portion of y, again using the exponential of the twist. First, we find the twist corresponding to the product of the Kalman gain K and the measurement residual y:

$$\begin{aligned} (Ky)_\phi &= (Ky)_q \otimes q^* \\ \Omega(Ky, v) &= ((Ky)_\phi, v \times (Ky)_\phi + (Ky)_v) \end{aligned} \tag{24}$$

Then, we integrate estimated pose using the exponential of this twist:

$$(x_{(q,v)})_{k|k} = \exp\left(\frac{\Delta t}{2}\Omega\right) \otimes (q, v) \tag{25}$$

Finally, the velocity component of innovation y is scaled and added:

$$(x_{\omega,\dot{v}})_{k|k} = x_{\omega,\dot{v}} + (Ky)_{\omega,\dot{v}} \tag{26}$$

3.4 Registered Visual Servoing

We use the computed camera registration ${}^{B}\mathcal{S}_{C}$ to servo to a target object according to the control loop in Fig. 2. This is position-based visual servoing, incorporating the dynamically updated registration. First, we compute a reference twist ${}^{B}\Omega_{e,ref}$ from the position error using camera pose ${}^{B}\mathcal{S}_{C}$ and object pose ${}^{C}\mathcal{S}_{o}$:

$${}^{B}\mathcal{S}_{e,ref} = {}^{B}\mathcal{S}_{C} \otimes {}^{C}\mathcal{S}_{obj} \tag{27}$$

$${}^{B}\Omega_{e,ref} = \ln\left({}^{B}\mathcal{S}_{e,act} \otimes {}^{B}\mathcal{S}_{e,ref}^{-1}\right) \tag{28}$$

Then, we find the reference velocity for twist ${}^{B}\Omega_{e,ref}$:

$$\begin{bmatrix} \dot{x} \\ \omega \end{bmatrix} = \begin{bmatrix} \mathbb{D}({}^{B}\Omega_{e,ref}) - (2\mathbb{D}({}^{B}\mathcal{S}_{e}) \otimes \mathbb{R}({}^{B}\mathcal{S}_{e})^{-1}) \times \mathbb{R}({}^{B}\Omega_{e,ref}) \\ \mathbb{R}({}^{B}\Omega_{e,ref}) \end{bmatrix} \tag{29}$$

where $\mathbb{R}(X)$ is the real part of X and $\mathbb{D}(X)$ is the dual part of X.

Finally, we compute joint velocities using the Jacobian damped least squares, also using a nullspace projection to keep joints near the zero position:

$$\dot{\phi}_r = J^+ \left(-k_x \begin{bmatrix} \dot{x} \\ \omega \end{bmatrix} \right) - k_\phi (J^+ J - I)\phi \tag{30}$$

where J is the manipulator Jacobian matrix, J^+ is its damped pseudoinverse, k_x is a gain for the position error, and k_ϕ is a gain for the joint error.

4 Experiments

We implement this approach on a Schunk LWA4 manipulator with SDH end-effector, see Fig. 1, and use a Logitech C920 webcam to track the robot and objects. The Schunk LWA4 has seven degrees of freedom and uses harmonic drives, which enable repeatable positioning *precision* of ±0.15mm [7]. However, absolute positioning *accuracy* is subject to encoder offset calibration and link rigidity. In practice, we achieve ±1cm accuracy when using only the joint encoders for feedback. The Logitech C920 provides a resolution of 1920x1080 at 15 frames per second. To measure ground-truth distances, we used a Bosch DLR165 laser rangefinder and a Craftsman 40181 vernier caliper.

We initially test the convergence and resistance of our approach while moving the camera. With the camera mounted on a tripod, we compute the filtered registration while the camera is perturbed, rotated, and translated.

The resulting registrations under moving camera are plotted in Fig. 4. The visual pose estimates contain frequent outliers in addition to a small amount of noise. The filtered registration removes the outliers and converges within 5s.

To demonstrate the suitability of this approach for manipulation tasks, we test the positioning accuracy attainable with this online registration. As shown in Fig. 5, we place a marker on a table, measure linear distance to the marker with a laser ranger, servo the end-effector to the visually estimated marker position using the control loop in Fig. 2, and measure the distance to the end-effector which should be directly over the marker.

The resulting position accuracy achievable with online registration is summarized in Table 1. For an ideal camera placement with close, direct view of the end-effector (i.e. the angle δ between the camera and the markers is 45° or less), positioning accuracy is in the submillimiter range. Larger camera distances and angles, resulted in positioning error of $1 - 2$ millimeters.

Finally, we test the pre-grasp positioning accuracy of this method as shown in Fig. 6. We place an object, in particular, a cup, at a variety of locations on the table, servo the end-effector to the visually detected object position using the control loop in Fig. 2, and then measure the distance of each finger to the object using a vernier caliper.

The results of the pre-grasp positioning are summarized in Table 2. A small number of trials resulted in centimeter-level error for objects placed near the edge of the image frame. Ommitting these outliers, the average positioning error of the pre-grasp configuration was 3.3mm.

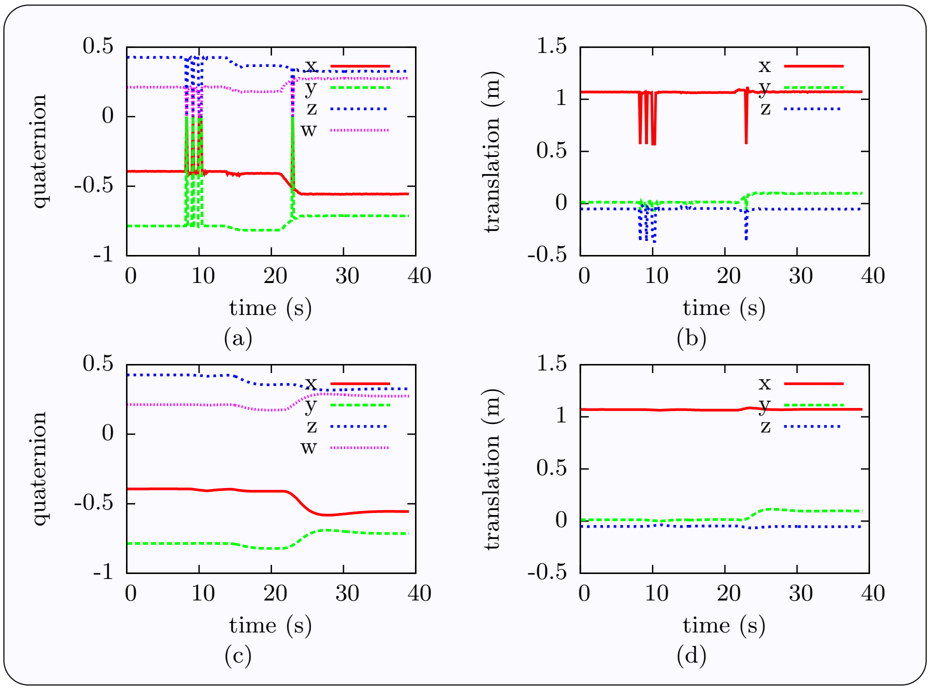


Fig. 4. Registration while camera is bumped (8s), rotates (15s) and translated (24s). camera is bumped. (a)-(b) registration from raw visual pose estimates of one feature. Contains many outliers. (c)-(d) filtered registration. Outliers and noise eliminated.

5 Experimental Insights

There are a number of error sources we must handle in this system. For the kinematics, error from encoder offsets in the arm, imprecise link lengths, and flexing of links all contribute inaccurate kinematic pose estimates. For perception, error from inaccurate camera intrinsics, imprecise fiducial sizes, offsets in object models, and noise in the image all contribute to error in visual pose estimates. To achieve accurate manipulation, we must account for these potential sources of error.

The key point of the servo loop in Fig. 2 is that we depend not on minimizing *absolute* error, but on minimizing *relative* error. We are minimizing error between end-effector pose $\mathcal{S}_e$ and target pose $\mathcal{S}_o$. Because we continually update the camera registration, we effectively minimize this error in the image. As long as there is distance between camera frame poses ${}^{C}\mathcal{S}_e$ and ${}^{C}\mathcal{S}_o$, we will move the end-effector towards the target, and as long as the visual distance estimate is zero when we reach the target, the arm will stop at the target. Thus, even if there is absolute registration error due to, e.g., unmodeled lens distortion, it is only necessary that relative error between visual estimates of the end-effector and target be small and converge to zero. The relative error between end-effector and

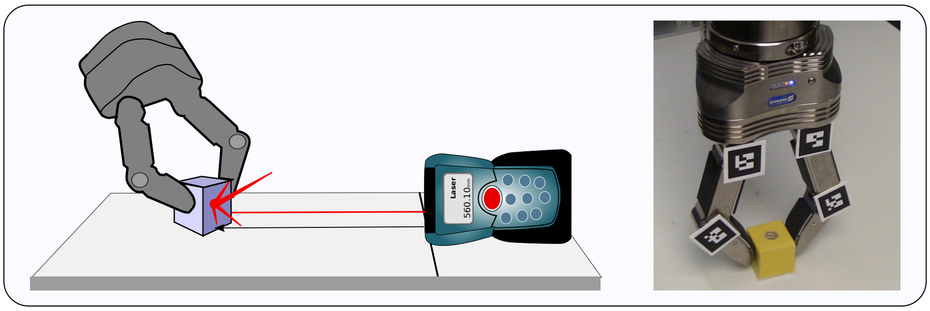


Fig. 5. Experimental setup for evaluating the positioning accuracy during camera registration. A cube was placed on a marker and the distance to a laser ranger was captured. Subsequently, the cube was placed in the hand of the robot, which, then, servoed to the position of the marker. Again, the distance was measured using the laser ranger.

target is crucial in manipulation, and our technique is well suited to minimizing this error.

The position of the tracked features on the robot has an important effect on error correction. Kinematic errors between the robot body origin and the tracked features, e.g., due to flex or encoder offsets, are incorporated into the camera registration and handled through the servo loop. Error between the observed features and the end-effector cannot be corrected. Thus, it is better to track features as close to the end-effector as possible. Consequently, we placed the fiducual markers on the fingers of the SDH end-effector.

The principal challenge in the implementation stems from observing the robot pose using small, ≈3cm, markers. While marker translation is reliably detected, outliers in orientation are frequent. Ample lighting improves detection but does not eliminate outliers. The median pose, (4)-(6), was effective at eliminating outliers from visual estimates. Alternative methods for combining orientations estimates include Davenport's q-method [14] and the Huber loss function [9]. In contract to these other methods, the median has no parameters such as thresholds which require adjustment. Thus, it is especially suited to this online registration application where outlier frequency may vary depending on camera placement, lighting, etc. A potential challenge is that the direct computation of

Setup	Average	Stdev
$\delta \leq 45°$	0.5mm	0.52mm
$\delta > 45°$	1.5mm	1.26mm

Table 1. Positioning experiment results. Average and standard deviation [mm] of measured difference between commanded position and object location.

Data	Average	Stdev
All	5.8mm	8.5mm
Inliers	3.3mm	2.3mm

Table 2. Pre-grasp experiment results. Average and standard deviation [mm] of measured difference between object and end-effector position

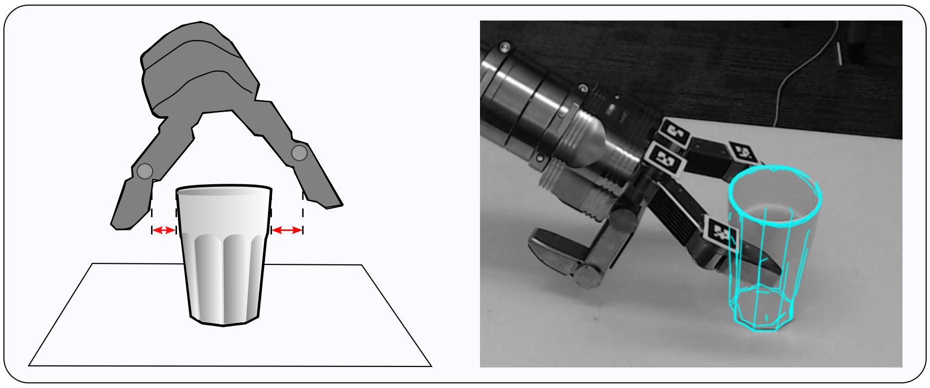


Fig. 6. Pre-grasp experiment: using the introduced camera registration, the open robot hand is servoed to the position of a glass. The distances between the fingers and the glass are then measured. Since the glass is rotationally symmetric, the distances of both used robot fingers should be identical in the ideal case.

(4) leads to an $O(n^2)$ algorithm in the number of orientations. However, for the small number of poses we consider at each step here, the computation time is negligable. On a Xeon E5-1620 CPU, computing the median of 32 orientations requires 30µs.

6 Conclusion

We have presented an online method to identify the camera poses for robot manipulation tasks. This is useful for the typical case where camera registration is not static but changes due to model error, disturbances, or wear and tear. The key point is to track both the object and the robot in the image, and servo based on the visually estimated relative pose between the object and robot. By combining median and Kalman filtering of the registration pose, we are able to achieve millimeter-level manipulation accuracy. We have shown in our experiments that online registration can be used to improve positioning accuracy during grasping and manipulation tasks, thereby avoiding typical challenges such as premature contact between fingers and objects.

A useful extension to this work would be to handle online registration with multiple cameras. This could provide additional data to improve accuracy or permit greater field of view, e.g., observing both hands in bimanual tasks. We anticipate that considering median deviation and applying a similar extended Kalman filter to multiple simultaneous poses will extend this online approach to multi-camera setups.

Acknowledgements

This work would not have been possible without Mike Stilman's tireless guidance and support.

A Dual Quaternion Computation

Dual quaternions are a numerically convenient representation for Euclidean transformations, $\mathcal{SE}(3)$. Compared to ordinary quaternions which can represent rotation, dual quaternions can represent both rotation and translation. Mathematically, they are the extension of quaternions to the dual numbers [18]. Dual numbers are of the form $r + d\varepsilon$, where r is real part, d is the *dual* part, and ε is the dual element such that $\varepsilon \neq 0$ and $\varepsilon^2 = 0$. A dual quaternion $\mathcal{S}$ can be represented as a pair of quaternions, $\mathcal{S} = s_r + s_d\varepsilon$, which we represent with the tuple $\left(s_r,\ s_d\right)$.

We represent the vector and scalar components of the ordinary quaternion parts of a dual quaternion as:

$$\begin{aligned} \mathcal{S} = \left(r,\ d\right) &= \\ \left(\left(r_x i + r_y j + r_z k,\ r_w\right),\ \left(d_x i + d_y j + d_z k,\ d_w\right)\right) &= \\ \left(\left(r_v,\ r_w\right),\ \left(d_v,\ d_w\right)\right) & \end{aligned} \tag{31}$$

where r_v and d_v are the vector parts and r_w and d_w are the scalar parts.

The dual quaternion representing orientation q and translation v is:

$$\mathcal{S} = \left(s_r,\ s_d\right) = \left(q,\ \frac{1}{2} v \otimes q\right) \tag{32}$$

Dual quaternion Euclidean transforms are normalized by dividing by the real magnitude:

$$S' = \left(\frac{s_r}{|s_r|},\ \frac{s_d}{|s_r|}\right) \tag{33}$$

Operations on the dual quaternions can be derived from those of ordinary quaternions and the properties of dual numbers. However, this requires care to handle singularities. Generally, the values at these singularities can be computed by identifying singular factors with convergent Taylor series. Given suitable singular factors, a computer algebra system, e.g., Maxima, Mathematica, can be used to compute the Taylor series. We summarize the relevant functions and suitable Taylor series below.

Dual quaternion multiplication is:

$$\mathcal{A} \otimes \mathcal{B} = \left(a_r \otimes b_r, \quad a_r \otimes b_d + a_d \otimes b_r\right) \tag{34}$$

The dual quaternion exponential is:

$$\phi = |r_v| \tag{35}$$

$$k = r_v \cdot d_v \tag{36}$$

$$e^{\mathcal{S}} = e^{\tilde{w}} \left[\left(\frac{s}{\phi} r_v, c \right), \left(\frac{s}{\phi} d_v + \frac{c - \frac{s}{\phi}}{\phi^2} k r_v, -\frac{s}{\phi} k \right) \right] \tag{37}$$

where $s = \sin\phi$, $c = \cos\phi$, $\tilde{w} = r_w + d_w \varepsilon$, and $r_v \cdot d_v$ is the dot product of r_v and d_v.

Then, to handle the singularity at $\phi = 0$, we use the following Taylor expansions:

$$\frac{\sin\phi}{\phi} = 1 - \frac{\phi^2}{6} + \frac{\phi^4}{120} - \frac{\phi^6}{5040} + \ldots \tag{38}$$

$$\frac{\cos\phi - \frac{\sin\phi}{\phi}}{\phi^2} = -\frac{1}{3} + \frac{\phi^2}{30} - \frac{\phi^4}{840} + \frac{\phi^6}{45360} + \ldots \tag{39}$$

The dual quaternion logarithm is:

$$\phi = \operatorname{atan2}\left(|r_v|, r_w\right) \tag{40}$$

$$k = r_v \cdot d_v \tag{41}$$

$$\alpha = \frac{r_w - \frac{\phi}{|r_v|} |r|^2}{|r_v|^2} = \frac{1}{|r|} \left(\frac{\cos\phi}{\sin^2(\phi)} - \frac{\phi}{\sin^3(\phi)} \right) \tag{42}$$

$$(\ln S)_r = \left(\frac{\phi}{|r_v|} r_v, \quad \ln|r| \right) \tag{43}$$

$$(\ln S)_d = \left(\frac{k\alpha - d_w}{|r|^2} r_v + \frac{\phi}{|r_v|} d_v, \quad k + \frac{r_w d_w}{|r|^2} \right) \tag{44}$$

where $(\ln S)_r$ is the real part of the logarithm and $(\ln S)_d$ is the dual part of the logarithm. Note that ϕ represents the angle between the real and imaginary parts of unit quaternion r.

To handle the singularity at $|r_v| = 0$ and knowing $|r| = 1$:

$$\frac{\phi}{|r_v|} = \frac{\phi}{\frac{|r_v|}{|r|}} = \frac{\phi}{\sin\phi} \tag{45}$$

$$\frac{\phi}{\sin\phi} = 1 + \frac{\phi^2}{6} + \frac{7\phi^4}{360} + \frac{31\phi^6}{15120} + \ldots \tag{46}$$

Then, for α in (42):

$$\frac{c}{s^2} - \frac{\phi}{s^3} = -\frac{2}{3} - \frac{1}{5}\phi^2 - \frac{17}{420}\phi^4 - \frac{29}{4200}\phi^6 + \ldots \tag{47}$$

References

1. H. Ben Amor, O. Kroemer, U. Hillenbrand, G. Neumann, and J. Peters. Generalization of human grasping for multi-fingered robot hands. In *Proceedings of the International Conference on Robot Systems (IROS)*, 2012.
2. François Chaumette and Seth Hutchinson. Visual servo control, part I: Basic approaches. *Robotics and Automation Magazine*, 13(4):82–90, 2006.
3. François Chaumette and Seth Hutchinson. Visual servo control, part II: Advanced approaches. *Robotics and Automation Magazine*, 14(1):109–118, 2007.
4. Changhyun Choi and Henrik I Christensen. Robust 3d visual tracking using particle filtering on the special euclidean group: A combined approach of keypoint and edge features. *The International Journal of Robotics Research*, 31(4):498–519, 2012.
5. Daniel Choukroun, Itzhack Y. Bar-Itzhack, and Yaakov Oshman. Novel quaternion Kalman filter. *Trans. on Aerospace and Electronic Systems*, 42(1):174–190, 2006.
6. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.
7. Schunk GmbH. Dextrous lightweight arm LWA 4D, technical data. http://mobile.schunk-microsite.com/en/produkte/produkte/dextrous-lightweight-arm-lwa-4d.html.
8. Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*, volume 114. John Wiley & Sons, 2011.
9. Peter J Huber et al. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.
10. Sungmoon Joo and Michael Grey. DRC-Hubo retrospective, January 2014. Personal Communication.
11. Matthew Klingensmith, Thomas Galluzzo, Christopher Dellin, Moslem Kazemi, J. Andrew (Drew) Bagnell, and Nancy Pollard. Closed-loop servoing using real-time markerless arm tracking. In *International Conference on Robotics And Automation (Humanoids Workshop)*, May 2013.
12. Joseph J. Laviola. A comparison of unscented and extended Kalman filtering for estimating quaternion motion. In *American Control Conference*, volume 3, pages 2435–2440. IEEE, 2003.
13. Ern J. Lefferts, F. Landis Markley, and Malcolm D. Shuster. Kalman filtering for spacecraft attitude estimation. *Journal of Guidance, Control, and Dynamics*, 5(5):417–429, 1982.
14. F Landis Markley, Yang Cheng, John Lucas Crassidis, and Yaakov Oshman. Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, 2007.
15. *OpenCV API Reference*. http://docs.opencv.org/master/modules/refman.html.
16. Vijay Pradeep, Kurt Konolige, and Eric Berger. Calibrating a multi-arm multi-sensor robot: A bundle adjustment approach. In *Experimental Robotics*, pages 211–225. Springer, 2014.
17. Kari Rainio and Alain Boyer. *ALVAR – A Library for Virtual and Augmented Reality User's Manual*. VTT Augmented Reality Team, December 2013.
18. Eduard Study. Geometrie der dynamen, 1903.
19. A. Teichman, S. Miller, and S. Thrun. Unsupervised intrinsic calibration of depth sensors via slam. In *Robotics: Science and Systems (RSS)*, 2013.

20. Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.