Learning Task Error Models for Manipulation

Peter Pastor∗, Mrinal Kalakrishnan∗, Jonathan Binney†, Jonathan Kelly†,
Ludovic Righetti∗‡, Gaurav Sukhatme†, Stefan Schaal∗‡
∗Computational Learning and Motor Control Lab, University of Southern California, Los Angeles, CA 90089, USA.
†Robotic Embedded Systems Laboratory, University of Southern California, Los Angeles, CA 90089, USA.
‡Autonomous Motion Department, Max-Planck-Institute for Intelligent Systems, 72076 Tübingen, Germany.

Abstract—Precise kinematic forward models are important for robots to successfully perform dexterous grasping and manipulation tasks, especially when visual servoing is rendered infeasible due to occlusions. A lot of research has been conducted to estimate geometric and non-geometric parameters of kinematic chains to minimize reconstruction errors. However, kinematic chains can include non-linearities, e.g. due to cable stretch and motor-side encoders, that result in significantly different errors for different parts of the state space. Previous work either does not consider such non-linearities or proposes to estimate non-geometric parameters of carefully engineered models that are robot specific. We propose a data-driven approach that learns task error models that account for such unmodeled non-linearities. We argue that in the context of grasping and manipulation, it is sufficient to achieve high accuracy in the task relevant state space. We identify this relevant state space using previously executed joint configurations and learn error corrections for those. Therefore, our system is developed to generate subsequent executions that are similar to previous ones. The experiments show that our method successfully captures the non-linearities in the head kinematic chain (due to a counter-balancing spring) and the arm kinematic chains (due to cable stretch) of the considered experimental platform, see Fig. 1. The feasibility of the presented error learning approach has also been evaluated in independent DARPA ARM-S testing contributing to successfully complete 67 out of 72 grasping and manipulation tasks.

I. INTRODUCTION AND RELATED WORK

Accurate modelling of kinematic chains is important for robots to successfully perform grasping and manipulation tasks. Therefore, a large body of research has been conducted to calibrate the kinematic parameters of robotic manipulators, see [1] for a comprehensive overview. The goal is to obtain very high precision for primarily industrial robots. In [2] and [3], the authors present methods to simultaneously calibrate the geometric parameters of the robot as well as the extrinsic and intrinsic parameters of the robot’s camera. Joint calibration of multiple sensors is considered in [4]. Similarly, work presented in [5] is inspired by the bundle adjustment approach, and generalized to estimate robot system parameters by including measurements from various types of sensors. In [6] the authors present an optimization approach that can make use of prior knowledge of the system. However, most of these methods only consider geometric parameters and therefore cannot account for potential non-linearities in kinematic chains. To achieve high accuracy even in the presence of non-geometric errors, e.g. due to gear transmission, friction, temperature, and compliance, research has been conducted towards also modelling these quantities and also optimizing those parameters, see [7] for an overview. However, accurate modelling of all possible non-geometric effects is tedious and potentially infeasible. In contrast, in [8], the authors propose to learn the body schema by determining the topology of the system through self-perception. Although this approach is interesting and ambitious, we argue that discarding the readily available forward model completely may harm accuracy and generalization. Instead we propose to account for non-geometric effects by learning a non-parametric error model on top of the existing (uncalibrated) forward model. Thus, our approach can be seen as a trade-off between having to manually model non-geometric effects and completely discarding the often-times readily available kinematic model. Essentially, using the analytical model achieves generalization across the entire workspace and the (local) error correcting model provides accuracy for the relevant part of the state space. This paradigm has been exploited for learning the dynamics model in [9]. In [10], the authors propose to learn the residual error using a three layered feedforward neural network. However, the experiments conducted consider a rather small workspace with limited non-linear effects achieving an improvement on the order of about 0.1 mm. In [11] the authors present their approach to account for the non-linearities present in their copy of the ARM-S robot. Their approach computes a position offset for the arm using k-nearest neighbors for different parts of the workspace. Orientation corrections are only computed for horizontal and vertical end-effector orientations. Furthermore, their approach cannot account for errors due to cable stretch given that two similar end-effector poses can have two very different pose errors depending on the configuration of the
arm, the particular reaching trajectory as well as its execution speed.

Especially in the context of manipulation, one might argue that arm/hand tracking and simultaneous object tracking by means of visual servoing is sufficient to achieve many of the considered tasks, even with uncalibrated cameras [12]. This holds true, nevertheless, we argue that an accurate initial estimate of the hand pose can only improve performance. Furthermore, precise forward models can be used whenever occlusions are present.

In this paper, we present a method to obtain accurate forward models even in the presence of non-linearities by learning the residual errors using Gaussian Process Regression (GPR). We argue that learning local task error corrections is sufficient if, and only if, the supporting planning and control architecture enforces subsequent executions to remain close to previous executions. It is very important to note, that this is not a limitation of the proposed approach, since the task error model considers the state space that is relevant for the task. We do not see any advantage of trying to achieve an accurate forward model for the entire state space. We want to stress that our approach can deal with non-linearities in kinematic chains, for example due to a counter-balancing spring or cable stretch. Calibrating the geometric parameters of the system as proposed in [2] cannot account such state dependent errors. Thus, our approach avoids this tedious calibration procedure altogether and learns (small) model errors together with the non-linear errors.

We want to note that the presented work has been applied in competitive, independent DARPA ARM (Autonomous Robotic Manipulation) testing and provided our system with enough accuracy to successfully perform 67 out of 72 task executions, see [13] for more details on the final results.

The remainder of the paper is structured as follows: A problem description is provided in II; The use of Gaussian Process Regression for model error learning is introduced in II-A and its applicability to manipulation is motivated in II-B; A system overview is provided in III; Experimental results on learning task error models for the head kinematic chain is presented in IV and for the arm kinematic chains in V; A conclusion and outlook for future work is presented in VI.

II. ACCOUNTING FOR NON-LINEARITIES IN KINEMATIC CHAINS

The goal of our approach is to learn error correcting models on top of existing forward models of our experimental platform, the ARM-S robot\(^1\), see Fig. 1. It is important to note that the considered kinematic chains include non-linearities that cannot be accounted for by aforementioned methods. The head is composed of two pan/tilt units mounted on top of each other. A spring passively counterbalances the head to support the under dimensioned motor of the top pan/tilt unit. This counterbalancing spring creates non-linear effects in the head kinematic chain. The non-linearities in the kinematic chains of the arms are due to cable stretch and the fact that the encoders are on the motor-side, see Fig. 2. Motor-side encoders obtain sensor readings directly at the motor as opposed to the joints, as such the cable stretch between the motor and the joints cannot be measured. The resulting error in both head and arm kinematic chains depend on the current joint angle configuration of the robot; Explicitly modelling these non-linearities seems difficult and tedious. The objective of calibrating the head kinematic chain, i.e. the two pan/tilt units, first is to enable our system to accurately merge 3D point clouds of perceived objects from different view angles. The objective of calibrating the arm kinematic chains is to get an accurate object pose estimate in the hand frame necessary for dexterous manipulation. For both these calibration tasks we propose to use the same two staged approach: First, optimize for the best transformation that minimizes the systematic pose error between the forward model and the visually perceived target pose. Second, learn a local non-linear correction term $\Delta x$ that absorbs the remaining task error as a function of the current joint angle configuration $\theta$.

A. Gaussian process regression for task error learning

We use Gaussian Process Regression [14] (GPR) to learn the remaining task error, i.e. the mapping $\theta \rightarrow \Delta x$. Gaussian Process Regression is particularly suitable because it can accurately fit the data and only very little tuning is required. The considered size of the data set is also well within the computational complexity limits of the standard GPR model. However, more advanced methods that use GPR are readily available for larger data sets [15].

The full 6D pose correction $\Delta x$ is learned using six separate GPR models, one for each task variable, i.e. translation in $x, y, z$ and corresponding rotations $\phi_{\text{roll}}, \phi_{\text{pitch}}, \phi_{\text{yaw}}$. The input to each of these models are the relevant $d$ joint angles.

\(^1\)Details about the considered experimental platform can be obtained at www.thearmrobot.com
\( \theta_i \in \mathbb{R}^d \). Let \( y \) be the scalar output variable that represents the correction for one of the six task variables in \( \Delta x \). Given a training set of \( n \) pairs \( \{ \theta_i, y_i \}_{i=1}^n \), the goal is to learn a function \( f(\theta^*) = y^* + \epsilon \) that predicts the correction term \( y^* \) for new joint configurations \( \theta^* \), where \( \epsilon \) is assumed to be additive noise with zero mean and noise variance of \( \sigma_{\text{noise}}^2 \). The observed targets \( y \) can be modeled by the Gaussian distribution \( \mathcal{N}(0, K(\theta, \theta) + \sigma_{\text{noise}}^2 I) \), where \( \theta \) is the set containing all joint angles \( \theta_i \), and \( K(\theta, \theta) \) denotes the \( n \times n \) matrix of the covariances evaluated for all pairs of training points \( \theta_i \) with all other training points \( \theta_j \) for \( i, j = 1..n \). A standard choice for the covariance function is the squared exponential covariance function \( k(\theta_i, \theta_j) = \sigma_{\text{signal}}^2 \exp(-\frac{1}{2}(\theta_i - \theta_j)^T W (\theta_i - \theta_j)) \), where \( \sigma_{\text{signal}}^2 \) denotes the signal variance and \( W \) the width of the Gaussian kernel. Predictions \( f(\theta^*) \) are obtained by conditioning the joint distribution of the observed targets \( y_i \) and the test target \( y^* \) on the observed inputs \( \theta_i \), the observed targets \( y_i \), and the query input \( \theta^* \), yielding

\[
f(\theta^*) = k(\theta, \theta^*) (K(\theta, \theta) + \sigma_{\text{noise}}^2 I)^{-1} y^* .
\]

Note that predicting \( f(\theta^*) \) can be computed within real-time control loops given that \( (K(\theta, \theta) + \sigma_{\text{noise}}^2 I)^{-1} \) can be computed beforehand. Evaluating the kernel function at the query point \( \theta^* \) and all other training points \( \theta \) results in negligible overhead. The hyperparameters \( \sigma_{\text{signal}}^2, \sigma_{\text{noise}}^2 \), and \( W \) of this Gaussian process remain the only open parameters which can be automatically estimated by maximizing the log marginal likelihood using standard optimization methods. See [14] for more information on Gaussian Progress Regression.

### B. Task error learning for manipulation

Our approach is tailored towards achieving accuracy in the region of the state space that matters. We argue that this is not a limitation of our approach because it perfectly fits our philosophy; We seek to employ stereotypical movements [16] that create stereotypical sensory events which are essential to detect failures [17] or allow for reactive behaviors [18].

To determine a set of good joint angle configurations that resemble the workspace of interest, we early on were logging all endeffector poses that required accurate kinematic calibration, e.g. pre-grasp poses, whenever the robot was performing a grasping or manipulation task\(^2\), see Fig. 1. Over the course of about 6 months we collected a total of 1786 such endeffector poses. These collected endeffector poses were converted into corresponding arm postures using an optimization based inverse kinematics (IK) method [19]. This method optimized over the nullspace postures and, in some cases, over a range of possible endeffector poses to find optimal joint angle configurations. Logging the endeffector poses as opposed to joint angles directly allowed us to reuse all stored IK requests, only requiring to recompute the joint angle configurations whenever we decided to adapt the cost function. Thus, the optimization based IK method tries to enforce that subsequent task executions will result in similar arm postures. Similarly, we employ an optimization based motion planning algorithm [20][19] to favor trajectories that are similar in shape and speed\(^3\). To obtain a realistic training set we used k-means clustering algorithm in joint space, where \( k \) was set to 150. Our results in Sec. V show that the selected set of joint angle configurations were suitable to cover the region of interest.

### III. System overview

In the following sections, we will describe how the presented approach is used to account for non-linearities in the head kinematic chain in Sec. IV and in the arm kinematic chains in Sec. V of the ARM-S robot. For both experiments we follow the same 3 staged approach. First, we generate a set of joint configurations that are relevant for the considered tasks and record these joint angles as well as the perceived 6D marker poses. Second, we use this data to optimize for a fixed transform that minimizes the pose errors between the forward kinematics model and the perceived 6D marker poses to account for systematic errors. Finally, we learn the remaining pose errors using GPR as described in II-A. Next, we will describe the stereo marker pose estimation method that has been used for both experiments.

#### A. Stereo marker pose estimation

The task error models are both learned with respect to the stereo camera, a Point Grey Bumblebee2 with 1024x768 pixel resolution. Prior to all experiments we calibrated for the intrinsic parameters. The target for both of the calibration procedures consists of target markers that have been designed to be easily detected by the ARToolKit\(^4\). The ARToolKit uses images from a single camera and infers the depth of using the a-priori known size of the detected markers. To increase the accuracy, especially in depth, we applied the ARToolKit to detect the corners of each marker in both images (obtained from the left and the right camera) and used the stereo calibration of the camera to compute the 3D corner position from the two 2D image coordinates. Finally, we average the 3D corner positions to obtain the center position of the marker. The orientation of that marker is computed by averaging the two obtained marker orientations. The full 6D marker pose is in the frame of the left camera. In addition to the pose of each marker \( m_i \), the ARToolKit also provides information about which marker \( i \) has been detected.

The target for the head calibration was composed of 4 rows with 8 distinct markers each arranged in a plane at a known distance, see Fig. 2 (left) and Fig. 3. This target has

\(^2\)A compilation of the considered grasping and manipulation tasks can be seen at http://youtu.be/VgKoX3RuvB0.

\(^3\)Internal testing using a VICON motion capturing system confirmed that the Barrett arm has a high repeatability, i.e. reaching for a particular joint configuration over and over again results in a small endeffector pose variance. However, varying the approach trajectory as well as the velocity increases the endeffector pose variance slightly. Nevertheless, most of the influence of the cable stretch still depends on the final arm posture.

\(^4\)ARToolKit (http://www.hitl.washington.edu/artoolkit) was originally developed by Dr. Hirokazu Kato, and its ongoing development is being supported by the Human Interface Technology Laboratory (HIT Lab) at the University of Washington, HIT Lab NZ at the University of Canterbury, New Zealand, and ARToolworks, Inc, Seattle.
been designed to cover the area of interest. The target for the hand-eye calibration procedure was composed of 4 distinct markers arranged as a cube, see Fig. 2 (right) and Fig. 5. These markers have been presented to the ARToolKit in a single object such that we could use the internal optimization routines and obtain a single transform. The final transform is obtained by averaging the transforms obtained from the left and right camera image.

IV. EXPERIMENT I: ACCOUNTING FOR NON-LINEARITIES IN THE HEAD KINEMATIC CHAIN

To cover the area of interest we selected 60 neck joint angle configurations for the two pan/tilt units, see Fig. 3. The lower pan/tilt unit allows to obtain (slightly) different views of a point of interest and can be used to avoid occlusions where possible. The selected joint angles consisted of the lower pan unit to be at \(-90^\circ, -45^\circ, 0^\circ, 45^\circ,\) and \(90^\circ\) and the lower tilt to be straight and all the way forward. For each of these \(5 \times 2 = 10\) configurations the upper pan/tilt joint angles were set such that the cameras point at 6 particular locations that cover the table. The actual experiment involved the robot moving to each of these 60 configurations and storing the 4 neck joint angles together with the visually perceived marker poses (using the method described in III-A) and their IDs as shown in Fig. 3. The IDs were used to recover the exact marker pose in the target frame shown in Fig. 2.

A. Optimizing for fixed transformations to account for systematic errors

First, we optimized for a fixed transformation from the upper tilt (UT) link to the left camera (LC) frame \(\mathbf{H}^t_{\text{UT}}\) to account for systematic errors. The transform from the base (B) frame to the upper tilt (UT) link is \(\mathbf{H}^t_{\text{UT}}\) and assumed to be correct. The 3D positions of the markers in the target (T) frame are denoted by \(\mathbf{p}_T\) and known since we designed the target. The 3D positions of the markers in the left camera frame are denoted by \(\mathbf{p}_{\text{LC}}\) and obtained using the method described in III-A. The 3D coordinates of the markers in base frame are denoted by \(\mathbf{p}_B\) and computed either by \(\mathbf{H}^B_T \mathbf{p}_T\) or by \(\mathbf{H}^B_{\text{UT}} \mathbf{H}^t_{\text{LC}} \mathbf{p}_{\text{LC}}\). The former requires knowledge about \(\mathbf{H}^t_T\) the latter requires knowledge about \(\mathbf{H}^t_{\text{UT}}\), see Fig. 2. Assuming a perfect model, i.e. correct transform from the base to the upper tilt link, and perfect perception, i.e. accurate sensing of the 3D marker positions in the left camera frame, then \(\mathbf{p}_B = \mathbf{H}^B_{\text{UT}} \mathbf{H}^t_{\text{LC}} \mathbf{p}_{\text{LC}} = \mathbf{H}^B_T \mathbf{p}_T\) for all detected markers. Thus, an error can be computed by \(\sum |\mathbf{H}^B_{\text{UT}} \mathbf{H}^t_{\text{LC}} \mathbf{p}_{\text{LC}} - \mathbf{H}^B_T \mathbf{p}_T|\). Our approach iteratively solves for these two transformations by minimizing this pose error keeping one fixed and solving for the other using a non-linear optimizer. Finally, this optimized transform \(\mathbf{H}^t_{\text{LC}}\) can be used to compute the remaining pose error for each of the 60 training configurations.

B. Learning a model to account for remaining error

To learn the remaining pose error we use Gaussian Process Regression (GPR) as described in II-A. We learn 6 GP models to predict the 6 correction terms, i.e. translation in \(x, y, z\) and corresponding rotations \(\phi_{\text{roll}}, \phi_{\text{pitch}}, \phi_{\text{yaw}}\). The input to each of the 6 GP models are all 4 joint angles of both pan/tilt units. Thus, the GP model provides us with a joint configuration depended 6 DOF pose correction which we apply to previously estimated fixed pose \(\mathbf{H}^t_{\text{UT}}\) to account for the remaining unmodeled non-linearities.

The use of a zero mean function for the GP ensures that the predicting correction term will be zero for regions of the state space that are far away from all training configurations. Thus, in the worst case the performance of our system will not degrade more than not using the GP correction, i.e. equivalent to using the forward model alone.

To evaluate the performance of our system we generated 60 random test configurations. These test joint angles are computed using inverse kinematics by pointing the cameras at uniformly sampled locations on the table. The joint angle configurations for the lower pan/tilt unit are restricted to be at one out of the 10 configurations used during training. Thus, we only consider test cases which we determined to be actually relevant for our grasping and manipulation tasks. Furthermore, we noticed that most of the non-linearities are due to the counterbalancing spring which mostly influences the upper pan/tilt, see Fig. 2. The primary objective when accounting for non-linearities in the head kinematic chain is the ability to perceive a particular object from many different view angles and obtain a small (ideally zero) object pose variation. Thus, the experiment consisted of accumulating all detected marker poses while moving through the 60 test configurations and computing the 1-standard deviation \(\sigma\) in position and orientation for each of the 32 markers. The obtained results, see Fig. 4, show that the GP correction successfully accounted for unmodeled non-linearities especially for position. The reduction in variance for the marker orientation is small due to the fact that small errors in the forward model will not cause huge errors in the perceived marker orientation. However, an error of \(2^\circ\) degrees can result in a positioning error of more than 4 cm. The statistics of the obtained results are shown in Tab. I.
trivial. Therefore, we propose to store previously executed pre-grasp and pre-manipulation poses while performing the task of interest and use a subset of these configurations to learn a task error model. As described in II-B, we collected a total of 1786 end-effector poses over the course of about 6 months. We used \( k \)-means clustering in joint space, i.e., after applying the optimization based inverse kinematics method [19], with \( k = 150 \) to obtain a representative set of task relevant arm postures. The 1786 recorded end-effector poses of the right arm have been mirrored and reused for the left arm\(^5\). The trajectories for these 150 arm postures for both arms have been computed using the optimization based motion planning algorithm [20]. To not alter the effect of cable stretch by replacing the Barrett hand with the marker cube (see Fig. 5) we constructed the cube to weigh exactly the same amount as the Barrett hand\(^6\). Due to varying lighting conditions, our marker pose estimation method described in III-A did not detect all 150 arm postures. We also did not take special care of filtering out arm postures that violate the visibility constraint and as such, only about 120 data points were collected.

A. Optimizing for a fixed transformation to account for systematic errors

To account for systematic errors in the mounting of the two Barrett arms with respect to the base frame, see Fig. 2, we optimized a full 6D transform. We used the same non-linear optimization technique to minimize the pose error obtained from the difference of the forward model and the visually perceived marker poses during training, similar to IV-A. The

\( 5\)We received the left arm only after reaching the second phase of the DARPA ARM-S project, see Fig. 1 (left). A video of the one armed version involved during the first phase can be seen at http://youtu.be/VgKoX3RavB0.

\( 6\)We decided to build a target cube as opposed to directly attaching markers to the Barrett hand primarily for convenience and to reduce the risk of misplaced markers at the remote test site during the ARM-S project.

V. EXPERIMENT II: ACCOUNTING FOR NON-LINEARITIES IN THE KINEMATIC CHAINS OF THE ARMS

Determining a set of arm postures that cover the relevant state space for the considered redundant manipulators is non-

![Resulting standard deviation after learning GP correction](image)

Tab. I. Summarized standard deviation over a total of 591 obtained marker poses from Fig. 4 for both cases, using forward kinematics only (FK) and using forward kinematics and GP correction (FK + GP). The results show the non-linear nature of the head kinematic chain (especially in Z direction) and that the GP correction significantly lowers the error.

![Screenshot of an example test configuration of the left arm](image)
obtained transform successfully accounted for the systematic error, as shown in Tab. II. The remaining non-linear error is computed using this fixed offset and stored together with the corresponding 7 joint angles of each arm respectively.

**B. Learning a model to account for remaining error**

To learn the remaining pose error we again employed 6 separate GP models and trained them using the corresponding 7 joint angles. The output of each of these 6 GP models are the corrections that absorb the unmodeled non-linearities. We evaluated the system by randomly choosing 150 joint configurations from the pool of 1789 previously executed and therefore task relevant arm postures. An example arm posture used during testing can be seen in Fig. 5.

The obtained result for the entire experiment for both arms can be seen in Fig. 6. The accumulated statistics are shown in Tab. II. The results show that the GP error correction significantly reduced the mean pose error as well as its variance. The obtained accuracy enabled the robot to accomplish precise grasping and manipulation tasks, see http://youtu.be/VgKoX3RuVBO. The source code used for the experiments is available at https://github.com/usc-clmc/usc-arm-calibration and https://github.com/usc-clmc/usc-clmc-ros-pkg. A longer (uncut) and high resolution (HD) version of the video supplement showing the two presented experiments can be seen at http://youtu.be/QKKViNGhWWQ.

<table>
<thead>
<tr>
<th>Error</th>
<th>x[cm]</th>
<th>y[cm]</th>
<th>z[cm]</th>
<th>roll[deg]</th>
<th>pitch[deg]</th>
<th>yaw[deg]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right arm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FK</td>
<td>1.6±0.8</td>
<td>1.4±1.0</td>
<td>1.2±1.1</td>
<td>1.5±1.1</td>
<td>2.1±1.7</td>
<td>5.1±1.7</td>
</tr>
<tr>
<td>FK+O</td>
<td>0.9±0.6</td>
<td>0.9±0.7</td>
<td>1.1±0.9</td>
<td>2.1±1.6</td>
<td>2.3±1.7</td>
<td>2.9±2.0</td>
</tr>
<tr>
<td>FK+O+GP</td>
<td>0.4±0.4</td>
<td>0.4±0.4</td>
<td>0.2±0.3</td>
<td>1.0±0.9</td>
<td>1.2±1.3</td>
<td>1.0±1.0</td>
</tr>
<tr>
<td>Left arm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FK</td>
<td>2.1±1.8</td>
<td>2.7±1.7</td>
<td>2.0±1.7</td>
<td>3.3±2.0</td>
<td>3.6±2.3</td>
<td>2.1±1.5</td>
</tr>
<tr>
<td>FK+O</td>
<td>0.8±0.9</td>
<td>1.0±0.8</td>
<td>1.0±1.0</td>
<td>1.9±1.3</td>
<td>1.8±1.6</td>
<td>1.4±1.1</td>
</tr>
<tr>
<td>FK+O+GP</td>
<td>0.4±0.5</td>
<td>0.4±0.4</td>
<td>0.2±0.3</td>
<td>0.8±1.0</td>
<td>0.9±1.0</td>
<td>0.6±0.6</td>
</tr>
</tbody>
</table>

Tab. II. Mean and 1-standard deviation of the results in Fig. 6. The pose errors are computed from the visually detected marker pose and the pose computed using the forward kinematics (FK), using forward kinematics including offset (FK+O), and forward kinematics including offset and GP correction (FK+O+GP). The results show the non-linear nature of the kinematic chain of both arms and that the GP correction significantly lowers the remaining pose error.

**VI. CONCLUSION AND FUTURE WORK**

We have proposed a kinematics calibration procedure that uses GP correction to account for non-linearities in kinematic chains. Our procedure does not need accurate kinematic calibration which cannot account for non-linearities in the first place. We want to stress the importance of stereotypical behaviors within our framework to ensure that subsequent executions remain close to previous experiences. Nevertheless, we want to emphasize that the presented data driven approach can easily be used to learn new tasks and new regions of
the state space. The feasibility of the presented approach has been validated in independent DARPA ARM-S testing. Both procedures presented in this paper have significantly contributed to the success in the final test.

In recent experiments, after this paper was accepted for publication, we realized an error in the upper pan unit, see Fig. 2 (right). After correcting for this (constant) error the initial calibration improved significantly. Nevertheless, we want to stress that we still employ the presented (unmodified) method to improve the accuracy of our system even further.

Future work will include considering the computed co-variance of obtained predictions as part of the cost function for finding good inverse kinematics configurations as well as motion plans, similar to [19]. We are also working towards using the obtained improved forward model to better initialize an arm/hand tracking algorithm to implement visual servoing.

ACKNOWLEDGEMENTS

We want to thank Jeremy Ma and Nick Hudson from the JPL ARM-S team for providing the intrinsic calibration methods of the Bumblebee2 stereo camera.

This research was supported in part by National Science Foundation grants ECS-0326095, IIS-0535282, IIS-1017134, CNS-0619937, IIS-0917318, CBET-0922784, EECS-0926052, CNS-0960061, the DARPA program on Autonomous Robotic Manipulation, the Army Research Office, the Okawa Foundation, the ATR Computational Neuroscience Laboratories, and the Max-Planck-Society.

REFERENCES
