Seeing Through Uncertainty: Robot Pose Estimation Based on Imperfect Prior Kinematic Knowledge

Leonard Klüpfel, Lukas Burkhard, Anne Elisabeth Reichert, Maximilian Durner, and Rudolph Triebel

***Abstract*—We present PK-ROKED, a learning-based pipeline for probabilistic robot pose estimation relative to a camera, addressing inaccuracies in forward kinematics, particularly in systems with elastic and lightweight modules. Our approach integrates a probabilistic 2D keypoint detection mechanism that leverages prior knowledge derived from the robot's imprecise kinematics. We further improve the detection accuracy and geometric understanding by incorporating segmentation of the robot arm. The method computes reliable uncertainty estimates, enabling a robust 2D-6D fusion for precise robot arm pose estimation from a single detected keypoint. PK-ROKED requires only synthetic training data, effectively exploits imperfect kinematics as valuable prior knowledge, and introduces a novel fusion framework for enhanced robot pose estimation. We validate our method on the Panda-Orb dataset, demonstrating competitive performance against state-of-the-art approaches. Additionally, we evaluate on two other robotic systems in real-world scenarios and show its practicality by using the predictions to initialize a tracking algorithm. Code and pre-trained models are available.**

***Index Terms*—Deep Learning in Robotics and Automation, Computer Vision for Other Robotic Applications, Sensor Fusion, Visual Tracking.**

I. Introduction

THE primary objective of robotic manipulation is to precisely interact with the target object. This requires knowledge about the pose of both the robot's end-effector and of the object that is supposed to be manipulated. Additionally, both need to be represented in a common reference frame. Usually, the end-effector pose is derived from forward kinematics and given relative to the robot's base frame. Whereas manipulation targets are estimated in the camera frame through dedicated perception algorithms, e. g., [1] or [2]. As the end-effector pose and the object pose are obtained w.r.t. different reference frames, connecting them forms an open kinematic chain: from the object to the camera, via the robot's base and to the end-effector. To align them in a common reference frame, an exact camera-to-robot-arm transformation is necessary, typically performed via hand-eye calibration.

As robotic applications expand from purely accuracy-related to a broader set of tasks, robots become more elastic and lightweight [3]. As a consequence, the kinematic chain from camera to the end-effector can be inaccurate with errors that are potentially dynamic, non-linear, or non-deterministic[1]. A static hand-eye calibration is unable to capture these dynamically changing errors which becomes a restricting performance factor and eventually prevents accurate and precise manipulation tasks. For instance, on the robot *neoDavid*, which features several elastic parts or nonlinear kinematic

Fig. 1. The humanoid robot *neoDavid* empties a dishwasher. In the bottom image, the error of the forward kinematics (magenta) of the robotic arm can be clearly seen, motivating a vision supported pose estimation (orange). The vision input is our PK-ROKED algorithm, which provides the image location and the associated covariance of selected keypoints (shown red and yellow in the insert, bottom left) for a subsequent sensor fusion. Image: ©DLR.

L. Klüpfel, L. Burkhard, A. E. Reichert, M. Durner, and R. Triebel are affiliated with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Wessling, Germany (e-mail: firstname.lastname@dlr.de).

R. Triebel is also with the Institute for Anthropomatics and Robotics, Intelligent Robot Perception, Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany.

[1]Non-deterministic in our context means that current real-time capable forward kinematic algorithms are unable to compute the result for the deformations accurately, given limited information on the load of the system.

 DOI: 10.1109/TRO.2025.3577030

components, causing a significant offset between the true robot arm pose and the pose obtained from kinematics, as it is illustrated in Fig. 1.

Given the outlined challenge, the question is how to achieve precise robotic manipulation with an inherently inaccurate robot? This boils down to correcting the erroneous end-effector pose estimate relative to the camera using online perception methods. Previous approaches trying to solve this challenge can broadly be divided into three categories *marker-based approaches* (discussed in Section II-A, e. g., [4]), *continuous tracker-based approaches* (see Section II-B, e. g., [5]) and *learning-based approaches* (details in Section II-C, e. g., [6]). However, each have disadvantages for the outlined situation. The first, requires markers to be fully visible in a given image to be able to establish a transformation. It furthermore requires the markers to be attached on the arm, restricting the robot's design.

Tracking-based approaches follow their target over consecutive frames, assuming locally limited changes in the target pose. Therefore, they require an initial pose estimate as initialization. This initial estimation is usually obtained through either of the other two previously mentioned categories. Alternatively, they are initialized by moving the robot into pre-defined configurations, which restricts the workflow of the robotic application. Tracking methods come at the disadvantage of potentially losing the tracked object due to fast motions, occlusions, or change of illumination conditions. In case of tracking and losing a robot arm, the robot's workflow has to be interrupted to obtain a new initial guess for a re-initialization.

Learning-based approaches circumvent the previous issues, which makes them a promising research direction. Interestingly though, the state-of-the-art mostly neglects other readily available sensor information besides camera images, i. e., the knowledge on the robot's kinematics. Although, this knowledge might be a potential error source, it is generally bounded within a range. Therefore, it can be of valuable information to indicate regions of interest or spatial relations, or act as a plausibility check for the computer vision results.

To this end, we propose a two-step robot arm pose estimation approach. First, we present our learning approach Prior Knowledge Robot Keypoint Detection (PK-ROKED)[2] which outputs 2D keypoints on a robotic arm and its binary segmentation mask. Our approach combines a single RGB image with *prior kinematic knowledge*. As the forward kinematics have a bounded error, we can leverage this valuable information for an improvement in our keypoint prediction and eventually our pose estimation. We do not rely on additional visual sensors (e. g., depth cameras), which enhances the applicability across various robotic systems and environments. In contrast to other methods, PK-ROKED does not only detect the keypoints' location in an image, but also predicts their corresponding uncertainty distribution such that its results can be combined with other information in downstream tasks. In this work, we present the fully developed method of PK-ROKED, which is based on our initial concept from [7]. As a second step, we correct the 6D robot arm pose using Bayesian filtering. There, we fuse a probabilistic model of the forward kinematics [8] – that is aware of the errors – with the output of the previous step.

Concretely, in this work we contribute the following:

- **Detection**: We propose our learning-based PK-ROKED approach, purely trained on synthetic data, which outputs 2D keypoint predictions and a binary robot arm segmentation.
- **Prior Knowledge**: We demonstrate how to leverage imperfect prior kinematic knowledge as a guideline for potential keypoint locations.
- **Probabilistic**: We compute meaningful predictions with corresponding uncertainties through leveraging Bayesian learning. This is of significant importance as an associated uncertainty allows us to combine these results with other sensor modalities for downstream tasks as e. g., fusion.
- **Single Keypoint Fusion**: We integrate our 2D keypoint detection into a fusion algorithm to provide one holistic framework to obtain estimated 6D robot poses, that requires only one single detected keypoint.
- **System Agnostic / Generalizibility**: We demonstrate the versatility and generalizability of our approach by applying it to two different robotic systems with varying complexity, in addition to a benchmark dataset. This is accompanied by a successful quantitative evaluation of the performance in the sense of accuracy, robustness, and precision.
- **Application**: We further show the applicability of our method on an advanced humanoid system in order to initialize a visual tracker as a means of bypassing the error-prone hand-eye transformation on an elastic robotic system.

We apply our approach primarily on two different robotic systems. Both have in common that an imprecise and inaccurate transformation between the respective camera and the end-effector restricts the system's manipulation performance: the

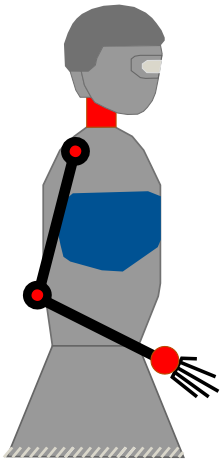


Fig. 2. Schematic visualization of mechanical sources of errors and uncertainties on the *neoDavid* system. Main errors stem from the elastic neck and arm.

[2]Available at: https://github.com/DLR-RM/PK-ROKED

wheeled humanoid *neoDavid* [9] and the planetary exploration prototype rover *LRU2* [10]. Both robots were developed at the Institute of Robotics and Mechatronics of the German Aerospace Center (DLR). Furthermore, we use the publicly available *Panda-Orb* dataset [6], to highlight our approach's competitive performance with the state-of-the-art in the topic of general keypoint detection and pose estimation for robots with accurate kinematics.

A. The Humanoid Robotic System: neoDavid

This system, as displayed in Fig. 1, has been designed to be particularly robust against mechanical impacts, e. g., collisions with its environment during or due to manipulation attempts. Additionally, an anthropomorphic appearance as well as human-like dexterous skills were further high priority design requirements. To this end, an elastic system was implemented through Variable Stiffness Actuators [11], which consist of springs and tendons. The main error sources specifically stem from a) a continuum-elastic neck [12], b) a non-linear wrist kinematic [13] and c) inaccurate measurements in the finger joints [8], as indicated in Fig. 2 with red error points. This leads to configuration dependent errors of several centimeters that can reach up to 20 cm for corner-case configurations. The resulting effect on the pose estimation is visible in Fig. 1. Clearly, one observes the large discrepancy between the orange skeleton, which represents our corrected pose estimation, and the magenta colored one, which is derived from forward kinematics with an additional neck estimation based on [12]– we refer to the resulting estimation as forward kinematics throughout this work if not stated otherwise.

B. The Rover System: LRU2

In case of the *LRU2* (short for Lightweight Rover Unit 2, shown in Fig. 3a), weight constraints have been primarily taken into account as it is a prototype robot for potential space applications. It is a four-wheeled rover which has a *Kinova Jaco2* arm mounted at its rear and a set of cameras located on a mast that allows for pan and tilt motions. The resulting key error sources are highlighted as red error points in Fig. 3b. The base plate of the robot's arm causes the main errors in the robot's forward kinematics due to a non-linear and load-dependent bending, especially when the arm is extended or grasps heavy objects. Furthermore, the joints of the robot arm contribute to the kinematic error, as they feature elasticities and the sensor readings are noisy. The final contributing error source is the base of the camera mast. Our own measurements yield resulting errors in the range of up to 2.5 cm between the assumed position of the end-effector as derived by forward kinematics, and its actual position.

II. Related Work

In this section, we present related work from three different categories: marker-based, continuous tracker-based and learning-based approaches.

A. Marker-based Approaches

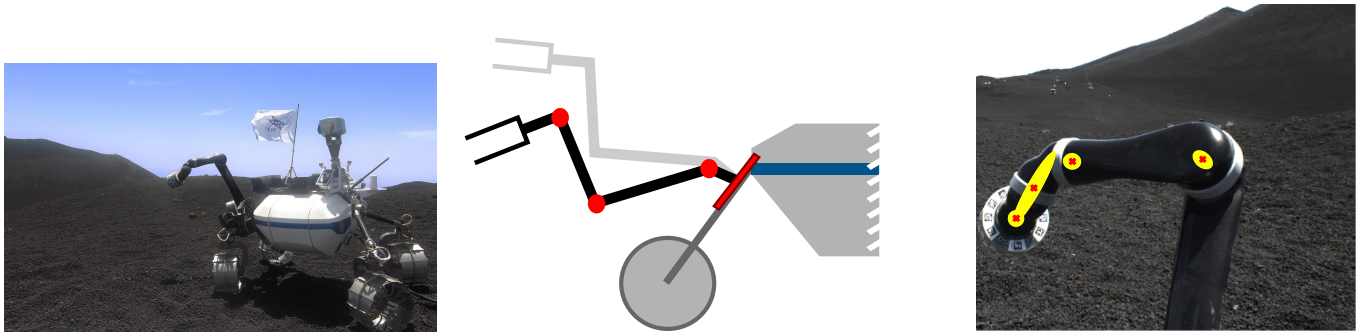
A classical approach to estimate the transformation between any object and a camera is by detecting fiducial markers, e. g., AprilTags [14] or ArUco [15], that are attached to the object. Placing these markers to a robot provides a relative 6D camera-to-robot transformation. This transformation together with the robot arm's forward kinematics and given joint angles can be used for the so-called hand-eye-calibration (e. g., the algorithm of Strobl and Hirzinger [16]) providing the correct transformation between the camera and the robot's base. Motivated by challenges such as forward kinematics being imprecise or an online re-calibration being required, Nissler *et al.* [4] introduce an algorithm which builds and optimizes a map of relative robot-to-marker transformations to derive the 6D camera-to-robot transformation. In [17], the authors use markers on the robot's arm for frequent and fast re-calibration of multiple sensors on the head of the humanoid robot. The marker-based approaches tend to be robust, but come with multiple downsides: attaching markers to a robot imposes constraints on both the design, as the markers need to be physically mounted, and also on the operation, as many marker types need to be fully visible for detection. Furthermore, the success and accuracy of the 6D pose estimation based on a single marker detection tends to correlate with the marker size. Our method differs to this category of approaches as it is markerless, and we can detect keypoints even with partial occlusions.

B. Continuous Tracker-based Approaches

This set of approaches to robot arm pose estimation are optimizing perceived data to match the closest point on a robot model and can be deployed to track a given robot [18]. Therefore, these methods rely in addition to visual sensory data - commonly depth data - on the forward kinematics and a geometric robot arm description as inputs. To track a robot arm, Schmidt *et al.* [19] model it as a kinematic chain of rigid bodies that is iteratively optimized in an adjusted Extended Kalman Filter (EKF) [20], while taking physical properties into account to facilitate grasping objects. In [21], the authors also model data input uncertainty in their approach to derive more robust estimations, considering both errors in the depth image but also in the joint encoders. Stoiber *et al.* [5] further introduce weighted constraints during the optimization process to combine the forward kinematics with estimates based on depth and RGB modalities. Disadvantageous to this group of methods is that good initialization values for the arm tracking are required and affect the overall performance. Furthermore, some of these approaches rely on depth data as a key modality - a sensory input that might yield robust and valuable data only in certain environments such as e.g., indoors. This is in contrast to our approach, that relies on RGB images as visual input and can detect as well as correct a pose in any robot arm configuration.

C. Learning-based Approaches

Learning-based methods differ in whether the 6D pose is directly derived or rather intermediary results are obtained, which are required for pose estimation. To this end,



(a) (b) (c)

Fig. 3. The LRU2 rover operating on the Volcano Mt. Etna: it is observing its arm motion using the head cameras (a), as the arm is subjected to several kinematic errors which are highlighted in red in the schematics (b). The real position of the arm is detected in the camera image, using our PK-ROKED method, providing us with the 2D keypoint position (red) and the associated covariances (yellow) (c).

Bohg *et al.* [22] introduce a Random Decision Forest classifier on depth images with an intermediary segmentation step. In order to obtain the actual joint positions for the robot arm position estimation, votes from pixels, which end up in the leaf nodes of the classifier, are clustered by relative joint offsets. This method is improved by directly regressing joint angles in their follow-up work [23].

Labbe *et al.* [24] approach pose estimation through iteratively rendering and comparing the image of a robot arm with its only input being an RGB image. The authors demonstrate that incorporating joint values as additional knowledge to the model improves the performance. Similarly, Ban *et al.* [25] also estimate the full 6D pose of a robot arm just from a single RGB image, without the joint parameters being known, as they argue this information may not always be accessible. Goswami *et al.* [26] derive the 6D pose in a similar setting and also investigate the impact of self-supervised pre-training on the performance.

To learn the hand-eye calibration, Chen *et al.* [27] apply differential rendering based on multiple RGB images as an input and optimize the joint parameters to estimate the 6D pose. In their follow-up work [28], the approach is extended to be training-free and thus robot arm agnostic by leveraging pre-trained segmentation models as supervisors for their training and prediction routine. A common approach of learning-based methods, which preliminary output intermediary results for robot arm pose estimation, is through first detecting keypoints in an RGB image of a robot [6], [29]–[32]. The usual next step is to forward these 2D locations with their corresponding 3D counterparts as obtained from forward kinematics to a Perspective-n-Point (PnP) [33] algorithm to derive the final 6D pose estimation [6], [32]. Improvements of this method are to optimize the set of keypoints [32], explicitly incorporating the robot arm's shape into the learning process [34], and back propagating the PnP error [35], [36]. The various approaches differ in whether they are purely trained on real-world data [29], on mixed datasets consisting of synthetic and real-world inputs [30], synthetic images only [6], [32], [37] or fine-tuned on real images in a self-supervised fashion [25], [26], [31], [35]. Tian *et al.* [37], [38] further expand this concept and formulate the problem statement as a tracking task with the pose from the previous estimations being optionally available to the current prediction. Our approach is similar to the methods of Lee *et al.* [6] and Lu *et al.* [32] but differs in a) incorporating prior knowledge from imperfect kinematics, b) the probabilistic learning formulation and c) Bayesian fusion instead of PnP being applied which comes at the advantage of being able to correct a pose given only one detected keypoint.

III. Method

Our proposed approach, Prior Knowledge Robot Keypoint Detection (PK-ROKED), consists primarily of two parts as displayed in Fig. 4:

- First, is our probabilistic 2D keypoint detection network.
- Second, follows our 2D-6D Bayesian pose fusion model.

At the first step, the robot keypoint detection network learns a representation of k predefined keypoints along a robot arm. In our work, a keypoint is a 2D projection of a known 3D point on the image plane. We take an RGB image $\boldsymbol{I} \in \mathbb{R}^{w \times h \times 3}$ with width w and height h as input. The second input is the 2D *prior kinematic knowledge* $\boldsymbol{P}_{\text{pk}} \in \mathbb{R}^{k \times 2}$ of our k defined keypoints which is discussed in detail in Section III-A1.

The output of our encoder-decoder model are k belief maps, one such map for each assumed keypoint location, guided by the prior kinematic knowledge, and additionally a segmentation mask of the robot arm (see Section III-A2 for details). To compute uncertainties for our keypoint predictions during inference, we perform t stochastic forward passes for a given input pair $\boldsymbol{I}$ and $\boldsymbol{P}_{\text{pk}}$ as outlined in Section III-A3. As an post-processing step, outliers can be filtered out through the predicted segmentation mask at inference time. The output of this step are up to k 2D keypoints $\boldsymbol{p}_j \in \mathbb{R}^2$ with associated covariances $\boldsymbol{\Sigma}_{2\text{D},j} \in \mathbb{R}^{2 \times 2}$ and a segmentation mask $\widehat{\boldsymbol{M}} \in \mathbb{R}^{w \times h}$, where j represents a keypoint that is not removed by our outlier rejection.

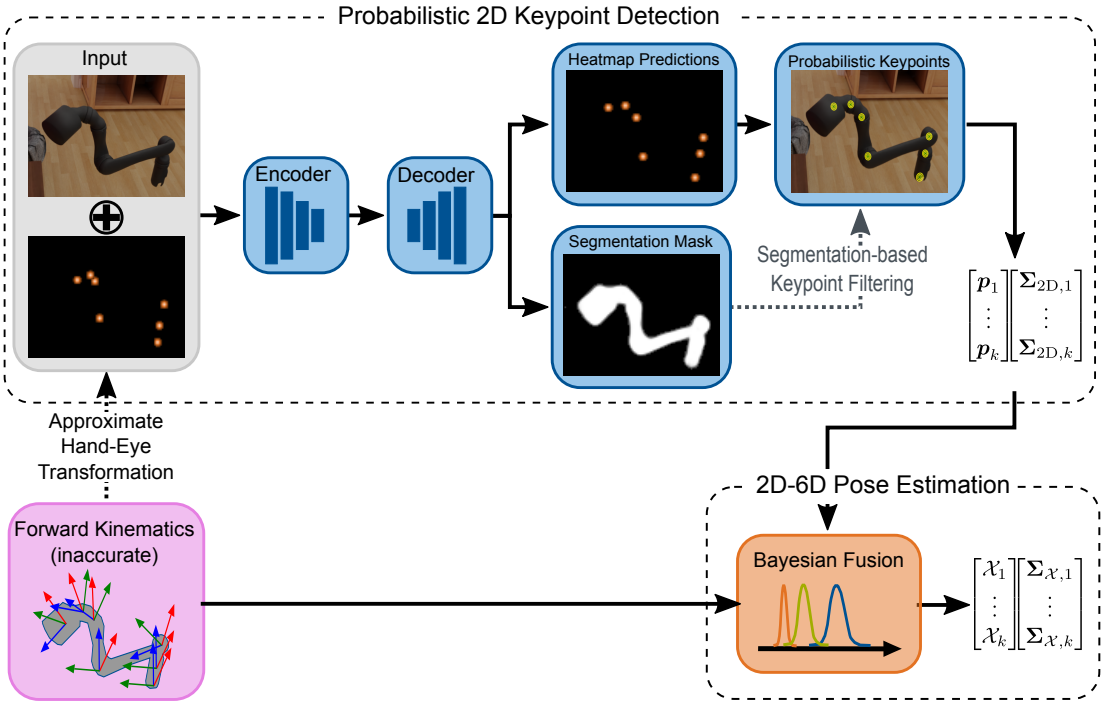


Fig. 4. Overview on Prior Knowledge Robot Keypoint Detection (PK-ROKED). First, at our probabilistic 2D keypoint detection step, we feed a single RGB image and prior kinematic knowledge to our encoder-decoder keypoint detection network. We output the mean coordinates and corresponding uncertainties $[\boldsymbol{p}, \boldsymbol{\Sigma}_{2D}]$ for our keypoint predictions in 2D image space, which we post-process through filtering based on a predicted robot arm segmentation mask. Second, we forward the obtained predictions to our 2D-6D pose fusion. At this step, we lift the 2D keypoints to 6D by fusing them with our probabilistic forward kinematics using a Kalman filter, such that we obtain a new pose estimate $\mathcal{X} \in SE(3)$.

At the second step, the 2D-6D fusion model, we lift the previously detected 2D keypoints to full 6D pose estimates that we represent as elements of the *special Euclidean group* $SE(3)$. To this end, this module takes the 2D detections and corresponding uncertainties $[\boldsymbol{p}_j, \boldsymbol{\Sigma}_{2D,j}]$ as well as predictions based on our probabilistic forward kinematics [8] as an input. Using an Extended Kalman Filter (EKF), mapping from 2D to 6D is achieved. The final output is the corrected 6D poses $\mathcal{X}_j \in SE(3)$ for up to k detected keypoints together with their covariance matrices $\boldsymbol{\Sigma}_{\mathcal{X},j} \in \mathbb{R}^{6\times6}$. In the following, we explain both steps in depth.

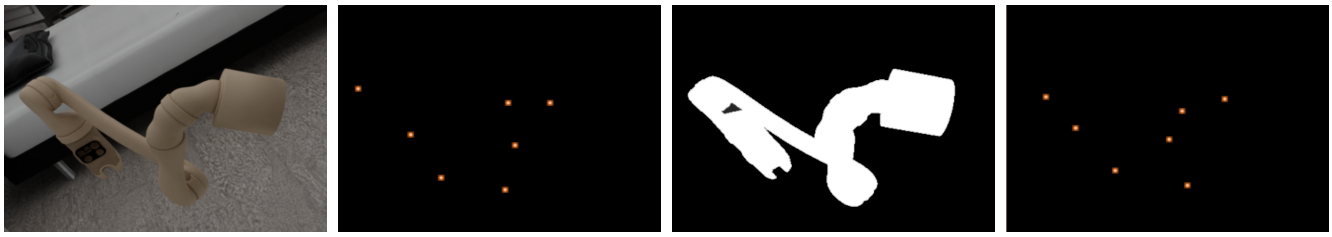
A. Probabilistic 2D Keypoint Detection

In this section, we present our deep learning approach to detect the 2D image keypoints. Note, that an initial version of this approach was introduced in [7], which we subsequently further improved for higher performance and robustness.

1) Prior Kinematic Knowledge: Our method assumes that we can derive the 2D image position of k keypoints for a given robot arm. As motivated however, these estimations are potentially subject to an error due to elastics in the kinematics or a hand-eye de-calibration and thus are inaccurate with a bounded error. Even though not perfectly precise, these erroneous estimations still provide valuable knowledge for a network by setting a focus area, in which the keypoints are likely to be. Therefore, we create for each keypoint one belief map, indicating the assumed location based on the erroneous forward kinematics. We concatenate the resulting k belief maps channel-wise to the image, such that the input to our network yields $w \times h \times (3 + k)$. The methodological framework to concatenate the assumed belief maps with the input image is characterized by its conceptual clarity and soundness. Furthermore, we incorporate the prior knowledge at almost no computational cost, as just minor changes in the encoder network are necessary, as explained in III-A2. In Fig. 5a and Fig. 5b we display both input components to our keypoint detection network. For visualization purposes, we depict the beliefs maps in one map in contrast to one map for each keypoint, as in Fig. 5b and Fig. 5d.

During training of our keypoint detection network, it is important to incorporate the prior kinematic knowledge without causing an overconfidence in this information. For this, we perturb the ground truth keypoint locations with Gaussian noise, such that: $\boldsymbol{P}_{\text{pk},i} \sim \mathcal{N}(\boldsymbol{P}_{\text{gt},i}, \sigma^2_{\text{pk, train}})$. $\boldsymbol{P}_{\text{gt},i}$ is the corresponding ground truth of each keypoint i, projected into the camera frame. We denote $\boldsymbol{P}_{\text{pk},i}$ as the 2D prior kinematic knowledge coordinates, sampled from the ground truth with $\sigma^2_{\text{pk, train}}$ as the variance.

The parameter $\sigma^2_{\text{pk, train}}$ represents the expected error in the prior kinematic knowledge. Thus, higher values of $\sigma^2_{\text{pk, train}}$ represent less precise forward kinematics of robotic systems. However, setting this parameter is only relevant during the training phase as the prior knowledge during the inference is already assumed to be imperfect and by this means perturbed. Note, that we explicitly use this independent sampling method for training to avoid the introduction of undesired correlations. We therefore refrain from directly using the a-priori probability distributions of the probabilistic forward kinematics [8] for



(a) (b) (c) (d)

Fig. 5. Data sample to train the *Jaco2* arm with a mounted docking interface to the end-effector: synthetic monocular RGB image is the first input component to our network (a) and perturbed prior kinematic knowledge as it is perceived by the network constitutes the second input component (b). The ground truth segmentation map is the training reference (c) together with the corresponding ground truth map with the actual keypoint locations (d). For visualization purpose, all keypoints in (b) and (d) are displayed in one map instead of one map per keypoint.

training and use this abstraction instead. To this end, we ablate the choice of this parameter and its influence on the performance of our method in Section V-D. Additionally to the magnitude of error in our prior knowledge, we indicate the region around a potential keypoint location by further applying Gaussian smoothening on the disturbed positions with $\sigma_{\text{smooth, train}}$. We associate a greater area around a keypoint as being more uncertain about the respective prior knowledge, whereas smaller areas analogously represent more certain areas. This parameter is set during the training and inference of our model. We ablate the importance of these two hyperparameters in Section V-D. By training with varying parameters and still benefiting from incorporating prior kinematic knowledge, we show that explicitly setting these two parameters is not required. Thus, exact knowledge about the uncertainty of a given robotic system is not crucial but only the magnitude of its errors. However, tuning these parameters yields the optimal performance and thus is still preferred if possible.

2) Encoder-Decoder Network Architecture: The encoder of our Convolutional Neural Network (CNN) is a ResNet50 [39] feature extractor pre-trained on ImageNet [40]. We adjust this backbone model to process $3 + k$ channels, which represent the image combined with the prior kinematic knowledge belief maps, by modifying the first convolution layer. The decoder consists of four upsampling blocks to decompress the learned representation. Each block comprises an upsampling operation (scale factor of 2), convolution layer, ReLU activation and another convolution (plus ReLU in the last two blocks only). For each task, predicting the location of k keypoints and segmenting the robot mask, we deploy one head of the same outline. These heads are built by three convolution layers with ReLU activations before and after the second layer.

To learn the keypoint locations, we train the network with a pixel-wise Mean Squared Error loss, $\mathcal{L}_{\text{kp},i}$, as:

$$\mathcal{L}_{\text{kp},i} = ||\boldsymbol{Y}_i - \widehat{\boldsymbol{Y}}_i||^2, \tag{1}$$

which directly measures the difference between the predicted, $\widehat{\boldsymbol{Y}}_i$, and ground truth, $\boldsymbol{Y}_i$, belief maps of the ith sample in the dataset. The ground truth labels are belief maps with a pixel value of one at the actual keypoint location and zeros elsewhere and smoothened around the keypoint edges with a Gaussian distribution of $\sigma_{\text{smooth, gt}} = 2$ pixels, as visualized in Fig. 5d. Consequently, non-visible keypoints, which do not fall into the camera frustum, have a zero valued belief map. To extract the actual keypoint coordinates, $\boldsymbol{y}_i^*$, from predicted belief maps, $\widehat{\boldsymbol{Y}}_i$, we find pixel peaks in the belief maps, after a Gaussian filtering as post-processing, similar to [6].

In order to learn the additional auxiliary task of segmenting the robot arm mask, we apply a Binary Cross Entropy loss, $\mathcal{L}_{\text{seg},i}$ as:

$$\begin{aligned}\mathcal{L}_{\text{seg},i} = \boldsymbol{M}_i \cdot \log(\mathcal{S}(\widehat{\boldsymbol{M}}_i)) + \\ + (1 - \boldsymbol{M}_i) \cdot \log(1 - \mathcal{S}(\widehat{\boldsymbol{M}}_i))\end{aligned} \tag{2}$$

with $\widehat{\boldsymbol{M}}_i$ as the mask prediction, $\boldsymbol{M}_i$ the binary mask ground truth and $\mathcal{S}(\cdot)$ being the Sigmoid function. In Fig. 5c, we show one ground truth binary mask, which we attempt to learn. The intuition to incorporate segmentation as an additional task is for the network to further gain a geometrical understanding of the robot arm which consequently benefits the main task of predicting keypoint locations. Eventually, the total loss for one data sample i, $\mathcal{L}_i$, derives as:

$$\mathcal{L}_i = \mathcal{L}_{\text{kp},i} + \lambda_{\text{seg}} \cdot \mathcal{L}_{\text{seg},i}, \tag{3}$$

where λ_{seg} defines a weighting factor. We train our keypoint detection network purely on synthetic images, for which we can trivially generate the required ground truth data.

3) Probabilistic Uncertainty Computation: One key contribution of our approach, next to incorporating prior kinematic knowledge, is computing uncertainties for our keypoint predictions as shown in [7]. To this end, we apply Bayesian learning to account for epistemic and aleatoric uncertainty in our approach, which are means to capture both the uncertainty over the model parameters and in the input data [41].

To model the epistemic uncertainty, we apply Monte Carlo dropout [42]. It approximates the intractable posterior distribution $p(\boldsymbol{W}|\mathcal{D})$ over the network parameters, $\boldsymbol{W}$, and a given dataset, $\mathcal{D}$, by minimizing the Kullback-Leibler divergence between a proxy distribution, $q_\theta(\boldsymbol{W})$, and the actual posterior distribution [41], [42]. It is further shown in [42] that this is equivalent to applying dropout [43] to the weight parameters during a network's training as well as running the inference with active dropout units. For an exact proof, we refer the reader to [42] and [41]. In our context, we implement Monte Carlo dropout by placing dropout units along the mid-layers of our encoder-decoder network as suggested in [44]. For

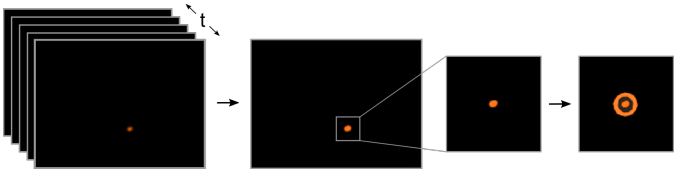


Fig. 6. Pipeline to compute the image moments based on t stochastic forward passes. First, belief maps per keypoint are stacked and accumulated. Next, the aggregated belief map is squeezed through a Sigmoid function and binarized at a threshold of 0.6. The resulting bright spot is a potential keypoint location, which we associate with the corresponding uncertainty of this detection. We capture the size and orientation of such regions by computing the image moments, which we can arrange in covariance matrices. The zoom in shows a visualization of the potential keypoint location and additionally with the resulting uncertainty ellipse computed based on the image moments.

inference, we perform t stochastic forward passes – with active dropout layers – where each pass can be interpreted as a Monte Carlo sample from the posterior distribution [45]. For each keypoint, the predictive mean $\boldsymbol{y}_j^*$ of our keypoint coordinates becomes:

$$\mathbb{E}(\boldsymbol{y}_j^*) = \frac{1}{t}\sum_{i=1}^{t} \boldsymbol{y}_i^* \tag{4}$$

with $\boldsymbol{y}_i^*$ denoting here the already extracted coordinates of the keypoints from the individual belief maps. This is our first output of the 2D step with $\boldsymbol{p}_j \coloneqq \mathbb{E}(\boldsymbol{y}_j^*)$.

We approximate the second order moment of the probability distribution, i. e., covariance, by applying the second order central image moment to predicted output heatmaps. In [7] we empirically demonstrated that modeling aleatoric uncertainty through these second order central image moments outperforms other such methods. The intuition and rationale is that the network output – the predicted belief maps about keypoint locations – indicates potential keypoints by higher pixel values in those maps. The wider and greater an image region is, the higher the uncertainty about the keypoint location. Consequently, we attempt to capture the size and orientation of these image regions about potential keypoints, as approximated by an ellipse. This approach further allows us to compute and directly measure the uncertainty in the belief maps. To compute these second order central image moments, we accumulate the belief maps of t stochastic forward passes per each keypoint, as shown in Fig. 6. Next, we process the belief maps by applying the Sigmoid function and thresholding the resulting map. Eventually, the image moments [46] are calculated as:

$$\boldsymbol{\Sigma}_{2\mathrm{D},j} \coloneqq \boldsymbol{\Sigma}(\boldsymbol{y}_j^*) \approx \begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix} \tag{5}$$

with

$$\mu_{nm} = \sum_{u,v} (u-\bar{u})^n (v-\bar{v})^m \mathrm{In}(u,v), \tag{6}$$

yielding the respective keypoints covariance matrix, which is the second output of the 2D computation step. For the image moment, the horizontal and vertical pixel coordinates are denoted by u and v, $\bar{u}$ and $\bar{v}$ are the image's centroid, and $\mathrm{In}(\cdot)$ is the pixel intensity value.

4) Segmentation based Outlier Rejection: For inference, before we further fuse our 2D keypoint detections to derive a 6D pose, we can post-process potential outlier keypoints through filtering based on our estimated robot arm segmentation. Therefore, we reject those keypoints that are not within our predicted segmentation mask. We process the singular masks from t forward passes by averaging and squeezing the result through a Sigmoid function and eventually binarizing it at a specific threshold. We note that this might be an useful step as it comes at no additional costs but with a potential practical upside.

B. From 2D to 6D: Robot Arm Pose Estimation

The outputs of our probabilistic keypoint detection network are the two dimensional image coordinates together with the associated uncertainty. However, most robotic applications require spatial information, which motivates a further 6D processing step. We present two ways to obtain the robot arm pose using the PK-ROKED 2D information. The first approach is used by the state-of-the-art, with several limitations. Due to this, we subsequently present our own approach based on sensor fusion, which is a further central contribution of this paper.

1) Perspective-n-Point: The state-of-the-art mostly uses algorithms for the well-known PnP problem to compute the camera-to-robot pose from the detected keypoints, for example [6] or [35]. For this, the 2D-to-3D correspondence of the keypoints is combined with the known camera intrinsics to estimate a camera-to-world transformation, where the world frame is usually located at the robot's base. There, the keypoints' 3D positions w.r.t. the robot's base are obtained from the forward kinematics.

Algorithms for the PnP problem are well established. However, they require a minimum of at least 4 keypoints (3 keypoints for some special variants) to be visible for a viable solution [33]. This restricts the camera perspectives from where the robotic manipulator can be observed. Furthermore, the method requires the robot manipulator to be a rigid body with precise joint measurements – the prerequisite to obtain the 3D coordinates in the world reference – a requirement that is fulfilled by industrial robots but not by the type of robots considered in this paper. Note that our method associates each measurement point with a covariance, which enables us to use probabilistic PnP methods like [47], a distinctive feature of our approach compared to all other keypoint detection algorithms.

2) Sensor Fusion on Lie Groups: We present a single shot sensor fusion, which relies strongly on the robot kinematics. Our approach only requires one detected keypoint from the network to obtain a valid pose, as it can leverage the kinematic information and the fact that kinematic errors act highly directional on the robot arm [8].

Our fusion approach is a single-step EKF-based state estimator that requires both a *prediction* and a *correction* input. We consider each keypoint individually, to assess the per-keypoint-accuracy of PK-ROKED in this paper. For each keypoint j, the keypoint's state $\mathcal{X}_j \in SE(3)$ is the pose of its reference frame w.r.t. the camera frame and has the associated uncertainty $\boldsymbol{\Sigma}_{\mathcal{X},j} \in \mathbb{R}^{6\times 6}$.

The state *prediction* is obtained from our probabilistic kinematics model [8] of the robot together with the current joint readings, which provides the priors $[\hat{\mathcal{X}}_j, \mathbf{\Sigma}_{\hat{\mathcal{X}},j}]$. The probabilistic robot kinematics allow encoding multiple errors in kinematic chains in a probabilistic manner. The method subsequently enables us to obtain a mean pose and the corresponding uncertainty of arbitrary points on the robotic structure – in our case the keypoints – by combining the configuration dependent influence of all errors w.r.t. the requested location. It is only necessary to define the magnitudes of the kinematic errors and their locations on the robotic structure. See [8] for details.

The detected 2D image coordinates of the PK-ROKED keypoints accompanied by the corresponding noise estimates constitute the measurements $[\boldsymbol{p}_j, \mathbf{\Sigma}_{\text{2D},j}]$ for the *correction*. The sensor measurement model is

$$\boldsymbol{z}_j = h(\mathcal{X}_j, \boldsymbol{v}_j), \tag{7}$$

where $\boldsymbol{z}_j \coloneqq \boldsymbol{p}_j \in \mathbb{R}^2$ and $\boldsymbol{v}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{\text{2D},j})$. As the detected keypoints are camera observations, we can use the well-known pinhole-projection model to rewrite (7) as

$$\boldsymbol{z}_j = \begin{bmatrix} \frac{t_1}{t_3} f_1 \\ \frac{t_2}{t_3} f_2 \end{bmatrix} + \boldsymbol{v}_j, \tag{8}$$

where, f_i is the corresponding focal length, and t_i denotes the ith translational component of $\mathcal{X}_j$.

The fusion is a one-step approach based on the correction step of an EKF that considers the Lie-Group structure of robotic poses. It is adapted from [48] to:

$$\boldsymbol{K} = \mathbf{\Sigma}_{\hat{\mathcal{X}},j} \boldsymbol{H}^T (\boldsymbol{H} \mathbf{\Sigma}_{\hat{\mathcal{X}},j} \boldsymbol{H}^T + \mathbf{\Sigma}_{\text{2D},j}), \tag{9}$$

$$\boldsymbol{m} = \boldsymbol{K}(\boldsymbol{z}_j - h(\hat{\mathcal{X}}_j, \mathbf{0})), \tag{10}$$

$$\mathcal{X}_j = \hat{\mathcal{X}}_j \operatorname{Exp}(\boldsymbol{m}), \tag{11}$$

$$\mathbf{\Sigma}_{\mathcal{X},j} = (\mathbb{I} - \boldsymbol{K}\boldsymbol{H}) \mathbf{\Sigma}_{\hat{\mathcal{X}},j}. \tag{12}$$

There, $\boldsymbol{K} \in \mathbb{R}^{6\times 2}$ denotes the Kalman gain, $\boldsymbol{H} \in \mathbb{R}^{2\times 6}$ represents the linearized observation matrix around $\hat{\mathcal{X}}$, $\boldsymbol{m} \in \mathbb{R}^6$ is the innovation term that resides in the local tangent space at $\mathcal{X}$ and is added to the prediction via the exponential map $\operatorname{Exp}(\cdot)$. Note that we use the vector representation of tangent space elements and our notation of the capitalized exponential map Exp implicitly contains the conversion of the vectors into locally valid Lie Algebras before transforming them into an element of the manifold $SE(3)$. For details on Lie Groups and the notation that we follow, we recommend [49].

IV. Evaluation Datasets

We test our algorithm in experiments on three datasets from different robotic systems. In each case, a robotic arm is observed by a camera system and the sensor data is recorded for algorithmic evaluation. These arms are Franka Emika's *Panda*, *LRU2*'s *Jaco2* and the right robotic arm of the humanoid system *neoDavid*. For the latter two, we record the datasets in our laboratory at DLR. The three systems differ in complexity, both in terms of design and uncertainties in their forward kinematics.

Our own recorded datasets have in common that they provide real camera based observations of the robot arm for inference, joint readings and the robot geometry for forward kinematics, a ground truth reference, and synthetic camera images for training.

A. Panda

To benchmark our approach, we evaluate on the publicly available *Panda-Orb* dataset, which is provided by [6]. This dataset contains 40k real-world images of the Franka Emika's *Panda* and additionally 100k synthetic images for training. The real-world data is recorded with a Realsense camera, placed at 27 different view points, while the arm performs the same motion for each camera position. As the public dataset contains several images with almost no motion between consecutive frames, we post-process the data to remove redundant configurations, and we eventually derive a dataset consisting of 32k images, which we evaluate on. This dataset is useful for comparability between our approach and other algorithms, as it is used for keypoint detection by e. g., [35] or [37]. To circumvent the missing forward kinematics and hence prior knowledge, which we require as an input, we utilize the ground truth and apply the same perturbation as during training. We argue that this dataset is the least complex and with the least uncertainties associated.

B. LRU2 and Jaco2 Arm

Second, we record data on the *Jaco2* arm mounted on the back of the *LRU2*. Generally, the rover is supposed to operate outdoors as shown in Fig. 3a. However, for our paper we record the data inside our laboratories to leverage the available VICON tracking system to obtain ground truth annotations. This provides us with a scene that has a cluttered background as shown in Fig. 15.

For the experiment, the arm moves to 29 different configurations, waits at each for five seconds, and resumes the motion to the next configuration, such that in total approximately 2k images are collected. The *LRU2* rover uses its Mako G319 camera together with a Schneider-Kreuznach Cinegon lens of 8 mm focal length to observe the motion of the robotic arm in a distance of $1-2$ m. The camera provides an RGB image with a resolution of 2064×1544.

We obtain the ground truth using the VICON tracking system. Two tracking targets are mounted on the rover, one on the pan-tilt camera head next to the RGB camera and one at the end-effector. The *LRU2* features a rotational-symmetric docking interface as end-effector. Attaching a tracking target there leaves us with one undefined degree of freedom for the end-effector ground truth. To overcome this issue, we use a specially designed VICON tracking target as shown in Fig. 15: the infrared tracking markers are attached to the borders of a small camera calibration plate [50] and the tool-adapter for the end-effector is mounted on the same calibration plate. This allows us to use well-established hand-eye calibration algorithms [16] to obtain the camera to robot-head-tracking-marker and the end-effector to end-effector-tracking-marker

transformations. Note that in this modified use of a hand-eye calibration, the robot kinematics part is replaced by the VICON-observed transformation between both tracking targets.

The subsequent VICON tracking provides us with a millimeter-accurate ground truth during the experiment. However, several loss-of-tracking events occur at specific robot arm configurations, when the VICON markers are occluded from the ceiling-mounted VICON-cameras. Those images without valid ground truth are excluded from the evaluation. Furthermore, our hand-eye calibration for the ground truth marker to end-effector pose remained with a residual rotation error of 2 degrees. This means that the positional ground truth measurement error increases further away from the tracking target, due to the lever that scales the rotational tracking error at the end-effector. We therefore limit our evaluation only to keypoints close to the end-effector, i.e., the last three joints, to avoid a potentially degraded positional ground truth. Furthermore, we deem this dataset and robot as of medium complexity and uncertainty.

In addition to the lab recorded data, we qualitatively perform tests on data collected on the volcano Mt. Etna during the ARCHES moon analog mission [51]. This is of great interest as the kind of images captured are a harsh contrast to our simulation and lab environment and are representative of the operation-environment for the *LRU2*. There exists no ground truth for this dataset so we are limited to visually inspect and judge the performance.

C. neoDavid Dishwasher

As final experiment, we use the dishwasher handling demonstration of our wheeled humanoid *neoDavid* as a real-world application scenario. In this sequence, *neoDavid* opens a dishwasher from which the robot subsequently removes a bowl and places it on a table that is located close by. Recall Fig. 1(top), where this scenario is shown.[3]

During the whole experiment, the robot's camera observes the arm, which allows our keypoint detection to run during the entire sequence. The camera is an Azure Kinect with RGB and depth capabilities, which the tracking algorithm takes both as an input. We capture approximately 8k RGB images at a resolution of 1280×720. For the 2D keypoint detection evaluation, we measure our performance on a subset of 3k images, as the motion between consecutive frames do not vary noticeable. Whereas for the pose estimation evaluation, we perform our pose estimation on all images.

Furthermore, we run the tracking algorithm of [5] during the task to use it as ground truth. The tracker uses both depth and color information of the robotic arm to track it based on the known model of the robot. The algorithm provides a precise estimation of the arm's pose as long as it is correctly initialized and no loss of tracking occurs. We initialize the tracking manually and monitor its performance to guarantee consistent and reliable tracking results. According to [52], the tracker can achieve sub-millimeter accuracy on idealized synthetic benchmark data. Examining our real world application, the tracker becomes more inaccurate due to model inaccuracies and sensor noise. An evaluation on the tracking data reveals stochastic errors mostly between $0.5\,\text{mm}$ and $2\,\text{mm}$ in magnitude with limited outliers reaching $10-20\,\text{mm}$.

[3] *neoDavid* emptying the dishwasher is also part of the video https://www.youtube.com/watch?v=N5QTvjFdIPM

V. Keypoint Detection Evaluation

In this section, we look into the 2D performance of the keypoint detection and compare it to the state-of-the-art.

A. Implementation Details

For all three robotic systems, we train on synthetic images only, (re-)sized $640 \times 480 \times 3$, which we generate for the *Jaco2* and *neoDavid* using BlenderProc [53] and with various domain randomization techniques [54]. We refer the reader to the Appendix A for a detailed list of the applied augmentations. For both arms, we roughly render 25k RGB images including segmentation maps. The Panda arm is trained on the publicly available synthetic dataset [6], consisting of approx. 100k images. For the latter dataset, we additionally generate binary arm masks, similar to [35]. For *Jaco2* and *neoDavid* we set the Dropout probability to $p_{\text{drop}} = 0.1$ and for the *Panda* to $p_{\text{drop}} = 0.2$ due to the different dataset sizes. The models for *Panda* and *Jaco2* are trained for 50 epochs, a prior knowledge training perturbation $\sigma_{\text{pk, train}} = 10$ pixels, $\sigma_{\text{smooth, train}} = 2$ pixels, and $k = 7$ keypoints represented by their joints. To train our approach on the *neoDavid* system, we set the epochs to 100 and $\sigma_{\text{pk, train}} = \sigma_{\text{smooth, train}} = 75$ pixels as this arm is the most challenging due to the least precise prior knowledge. Additionally, we select $k = 5$ keypoints along the right arm and hand. We refrain from selecting keypoints on the finger tips as their predictions turned out to contain a high amount of ambiguity. For all systems, we set $\lambda_{seg} = $ 1e-5 and apply AdamW [55] with a learning rate of $lr = $ 1.5e-4 and perform training on Nvidia RTX 3090 GPUs. Furthermore, as an effective batch size we choose 32 on *Jaco2* and *Panda*, whereas on *neoDavid*, we set it to 16. For all experiments, we evaluate with $t = 20$ forward passes over the same inference image. To filter outlier keypoints, we apply a binary threshold for our segmentation masks of 0.5, please note, this is applied for the 6D pose experiments, as in Section VII.

B. Metrics for 2D Evaluation

We assess the accuracy performance of our detections on the *percentage of correct keypoints (PCK)* metric in 2D image space [56]. *PCK* measures the Euclidean distance between the ground truth and our predictions at various pixel thresholds (*px*), which we refer to as *PCK@px*. Additionally, we only take visible keypoints into account. Higher values on this performance indicator are desirable. To evaluate our uncertainty computations and thus their meaningfulness, we apply the *Precision* metric. By checking whether a ground truth keypoint lies within the predicted uncertainty ellipse at varying scales, we can measure whether our detection is overconfident (low *PCK* accuracy combined with low ellipse multiples) or underconfident (high *PCK* accuracy and high

TABLE I
ACCURACY EVALUATION AS MEASURED BY *PCK* AT VARIOUS PIXEL THRESHOLDS. METHODS RELYING ON REAL-WORLD DATA ARE MARKED WITH *. THE VALUE MARKED WITH † REPRESENTS THE PERFORMANCE AT *PCK*@2.5 PIXELS.

Method	*Panda* *PCK* (@px) ↑					*Jaco2* *PCK* (@px) ↑					*neoDavid* *PCK* (@px) ↑				
	1	3	5	10	50	1	3	5	10	50	1	3	5	10	50
Ours	**0.068**	**0.491**	**0.793**	**0.944**	**0.992**	0.020	**0.189**	**0.428**	**0.786**	**0.98**	0.009	0.117	**0.277**	**0.428**	**0.727**
Ours w/ o seg.	0.05	0.436	0.742	0.922	0.991	**0.021**	0.176	0.425	0.768	0.957	**0.016**	0.103	0.212	0.32	0.569
Ours w/ o (PK + seg.)	0.05	0.404	0.686	0.822	0.844	0.016	0.118	0.25	0.427	0.66	0.012	**0.152**	0.260	0.418	0.610
DREAM w/ PK	0.057	0.398	0.679	0.871	0.969	0.012	0.087	0.244	0.569	0.828					
DREAM	0.041	0.35	0.631	0.766	0.789	0.004	0.019	0.047	0.12	0.224					
CtRNet*	0.096	0.635	0.922	0.996	1.0										
RoboPEPP*	[-]	0.28†	0.730	0.96	[-]										

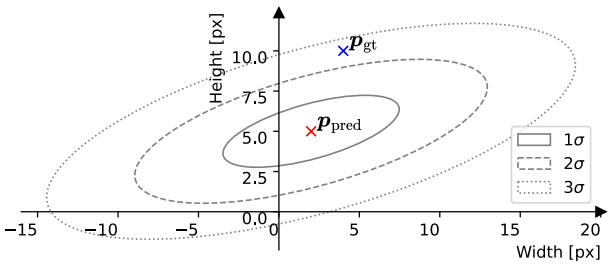


Fig. 7. Visualization of our uncertainty metric *Precision*, which illustrates our assessment with a toy problem. We check at which threshold s of a predicted keypoint, $\boldsymbol{p}_{\text{pred}}$, the actual ground truth, $\boldsymbol{p}_{\text{gt}}$, is enclosed.

ellipse scale multiples). As displayed in the toy problem in Fig. 7, the dummy keypoint prediction, $\boldsymbol{p}_{\text{pred}}$, and its corresponding uncertainty ellipse, at varying multiples, enclose the actual keypoint, $\boldsymbol{p}_{\text{gt}}$, only at a scale multiple of 3σ. Hence, the prediction would be regarded as True Positive (TP), otherwise False Positive (FP). Consequently, we compute for a given scale multiple threshold s:

$$\textit{Precision}@s = \frac{TP}{TP + FP}. \tag{13}$$

To check whether the actual keypoint is entailed by our uncertainty ellipse, we calculate:

$$r^2 = \frac{\boldsymbol{p}^2_{\text{gt, transf-x}}}{(0.5a)^2} + \frac{\boldsymbol{p}^2_{\text{gt, transf-y}}}{(0.5b)^2} \tag{14}$$

$$\mathbf{1}(r^2) = \begin{cases} 1, & \text{if } r^2 \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

with $\boldsymbol{p}_{\text{gt, transf}}$ denoting the ground truth keypoint relative to the ellipse center $\boldsymbol{p}_{\text{pred}}$ and transformed into this coordinate system. We perform the transformation with $\theta_{\text{transf}} = \arctan(\frac{v_{\text{max-y}}}{v_{\text{max-x}}})$, which expresses the uncertainty ellipse's orientation through v_{max}, the corresponding eigenvector of the eigenvalue $\lambda_{\text{eig, max}}$ of the detection's covariance matrix. Furthermore, $a = 2s\sqrt{\lambda_{\text{eig, max}}}$ is the ellipse's width along the major axis, $b = 2s\sqrt{\lambda_{\text{eig, min}}}$ the height of the ellipse along the minor axis and $\lambda_{\text{eig, max}}$, $\lambda_{\text{eig, min}}$ are the eigenvalues of the covariance matrix. The indicator function $\mathbf{1}(\cdot)$, in Equation 15, allows counting *TP* and *FP* uncertainty detections at a certain scale multiplier.

C. 2D Keypoint Evaluation Results

We quantitatively evaluate our approach on three datasets, one for each robot arm model, w.r.t. the previously introduced metrics. The presented results are given without the keypoint filtering step to better assess the actual prediction capabilities of our approach.

1) Results on Panda: On this commonly used dataset, we benchmark against our trained version of DREAM [6] and additionally we also compare against CtRNet [35] and RoboPEPP [26]. In Table I, we show that our approach achieves the highest performance when incorporating both segmentation and prior kinematic knowledge. This outperforms models using only prior kinematic knowledge or neither segmentation nor prior knowledge. Hence, we are already able to attest a positive effect of our prior knowledge approach on the detection accuracy. We further confirm this hypothesis as we outperform DREAM but can also improve this method, when we train it with our prior kinematic knowledge strategy. To this end, we slightly modify the architecture of DREAM to allow taking the additional prior knowledge belief maps as an input. By this we also highlight the generalizability and versatility of our approach, enabled by our simple and lean design choice of incorporating prior kinematic knowledge such that even existing approaches can benefit.
Benchmarked against CtRNet, we observe this method achieves higher accuracy results. In comparison to this approach, however, PK-ROKED trains only on synthetic data, trains shorter (for only 50 vs. 1000 epochs), does not require self-supervised learning on the real-world test dataset and additionally computes uncertainties for the predicted keypoints. We argue this performance gap can be attributed to the self-supervised fine-tuning on real-world data, for which CtRNet requires precise forward kinematics - an assumption we deem not to be valid for our robots.[4] However, to provide a comparison nevertheless, we ablate the performance possibilities of our approach in Section V-D, where we fine-tune on real-world data. Additionally, we observe that our synthetically trained model already outperforms RoboPEPP (trained on real-world data, self-supervised), which however assumes no knowledge about joint values at test time. The shown value at *PCK*@3 pixels for RoboPEPP actually represents their performance at

[4]We were not able to run CtRNet with the synthetic pre-training for a direct comparison with PK-ROKED and also were not able to train the approach for a new robot (e. g., our Jaco2), given the currently published code.

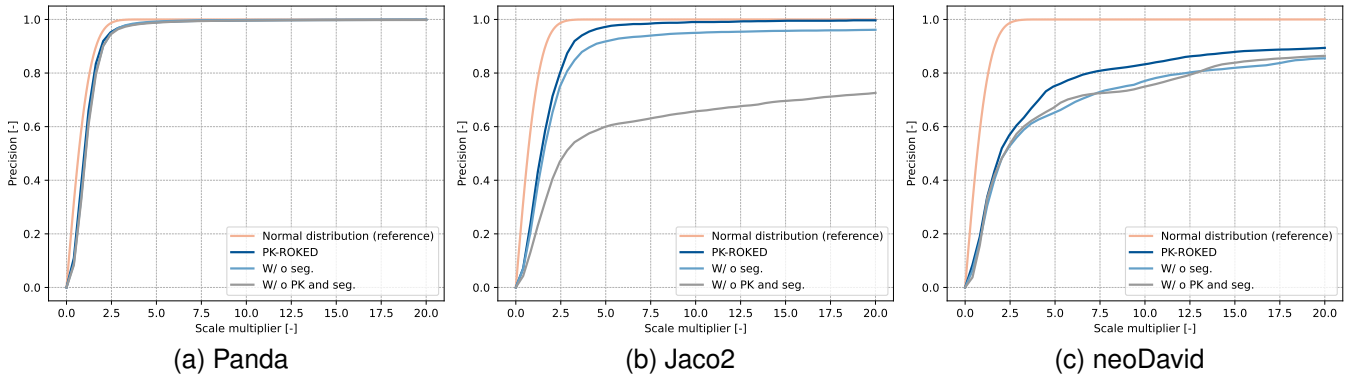


Fig. 8. Evaluation of our uncertainty computations as measured by *Precision*. On all three datasets, we observe our approach with binary segmentation as an additional task is best performing. This is indicated by higher *Precision* values being reached at smaller scale multiplies, i.e., the actual keypoints are captured by the computed uncertainty ellipse. On the *Panda* and *Jaco2* dataset, we deem our computed uncertainties as balanced due to the resemblance to the Normal distribution. Whereas on the *neoDavid* dataset, only uncertainties computed based on our approach with both, segmentation and prior knowledge, are to some extent able to be considered well-balanced between over- and underconfident.

their lowest reported *PCK* of 2.5, for which we achieve a score of 0.378 - also outperforming their approach.

On the *Precision* metric, as shown in Fig. 8a, we observe only a minor improvement on a nevertheless seemingly well-calibrated uncertainty computation. We base this on the similar - with only minor differences - trajectories of the respective *Precision* evaluations in comparison to the Normal distribution, which is regarded as balanced between under- and overconfident.

The qualitative results in Fig. 9, at the first column, verify our competitive results on this dataset. We observe mainly detections within a 2.5 pixel radius (green markers) to the ground truth (blue markers). In the few cases that are not within this range, we see that our predictions (red markers) and the respective uncertainty ellipse (yellow) capture the actual keypoints.

2) Results on Jaco2: On the medium complex and uncertain *Jaco2* dataset (Table I and Fig. 8b), we observe that a) our approach outperforms the DREAM method significantly and b) by incorporating the segmentation task, we enhance our performance on both metrics. Additionally, the impact of incorporating prior kinematic knowledge into a keypoint network becomes evident. This holds for both, the benchmarked approach DREAM and our method without prior knowledge and segmentation.

Furthermore, we are able to enhance our computed uncertainty predictions, which are deemed well calibrated due to the close resemblance of the Normal distribution. We interpret these findings twofold: first, the general benefit of prior kinematic knowledge for robot arm systems with uncertain pose estimations is apparent and second, we consider the segmentation as beneficiary to incorporate geometric reasoning as the accuracy as well as the *Precision* improves.

We confirm these findings qualitatively by looking at Fig. 9, at the second column. For visualization purpose, we show in these images all detected keypoints even though the quantitative evaluation takes into account only the last three from the end-effector. Our approach predicts keypoints (red markers) in immediate proximity to the actual ground truths (blue markers), which our uncertainty ellipse furthermore captures.

In addition to the lab recorded data, we show qualitative results on the ARCHES data in the third column of Fig. 9. We observe that our approach detects keypoints seemingly on the correct position even in this challenging environment. This is of special importance as it confirms that our approach provides reasonable results w.r.t. the *LRU2*'s operation environment.

3) Results on neoDavid: The arm and corresponding dataset are deemed the most challenging due to the high uncertainty in the prior kinematic knowledge and the complex mechanical system. On this dataset, we observe the greatest impact of our additional segmentation task. Without this, we actually have a worse accuracy performance compared to without prior knowledge, as displayed in Table I. We argue the underlying uncertainty incorporated through the prior kinematic knowledge is of such magnitude that it becomes challenging for the network to learn when and to what extent to rely on this prior knowledge. However, combined with the segmentation we again outperform on the higher *PCK* levels. Similarly, our approach that includes the segmentation achieves the best performance on the *Precision* benchmark and is the only one that comes close to the reference Normal distribution. We deduce this improvement can again be deemed as confirming our motivation to apply segmentation as enhancing the geometrical reasoning of our approach.

Looking at the last column of Fig. 9, we observe that our approach is detecting three (red markers) out of the four visible keypoints (blue markers) in the last image. It appears that this keypoint is especially challenging as there is also no computed uncertainty on the middle image. This occurs if the values in the belief maps of a keypoint are so low that in our processing step to compute the uncertainty ellipse the values are below the binarization threshold. Low pixel values in the belief maps indicate a low likelihood of a keypoint being at a certain pixel. Hence, we hypothesize that our network is uncertain about this keypoint. Nevertheless, our network detects the other keypoints consistently and their uncertainty ellipses capture the actual keypoints.

We interpret the results, observed on all three datasets

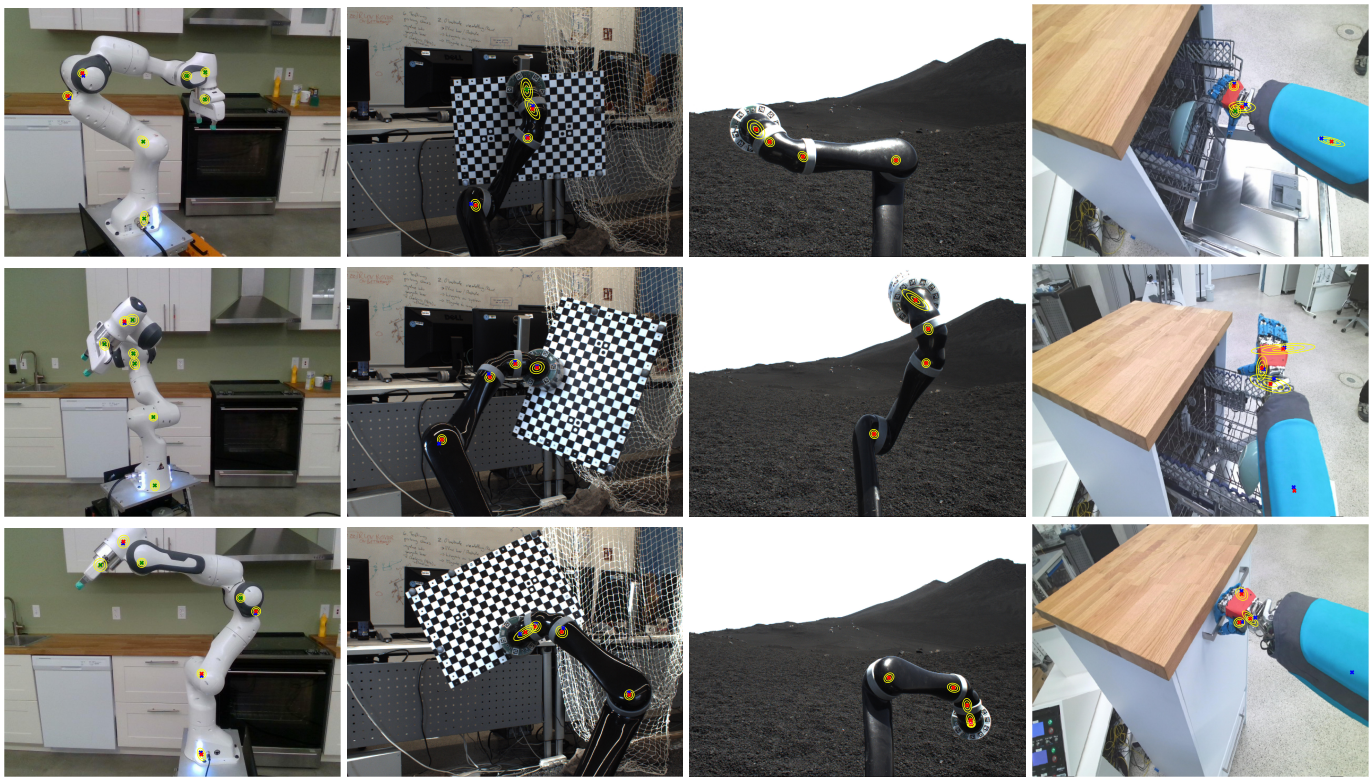


Fig. 9. Qualitative evaluation on all three robots. PK-ROKED detections are red markers, blue ones show the ground truth keypoint location. Green markers indicate keypoint detections with a 2.5 pixel radius to the ground truth. The yellow ellipse is our computed uncertainty. In the first column, we see our predictions on the *Panda* dataset. Most of our predictions are within the 2.5 radius and if that is not the case, our network captures the ground truth keypoints with the uncertainty ellipse - indicating well calibrated *Precision*. In the next columns, we show results on the lab as well as the more challenging ARCHES data for the *Jaco2*. Note that we are depicting all detected keypoints here, even though quantitative evaluation considers only the last three from the end-effector. For the lab data, we observe our predictions in very close proximity to the ground truth. Additionally, we capture the ground truth again with our uncertainty ellipse. For the ARCHES data, we immediately recognize the harsh contrast to the lab data. On the arm we can see challenging reflections and an overall stark contrast with the sky. Nevertheless, our network predicts seemingly correct keypoints with reasonable uncertainties. Eventually, we see results on the complex *neoDavid* dataset. In the last of those images, we see that one of the keypoints is not detected.

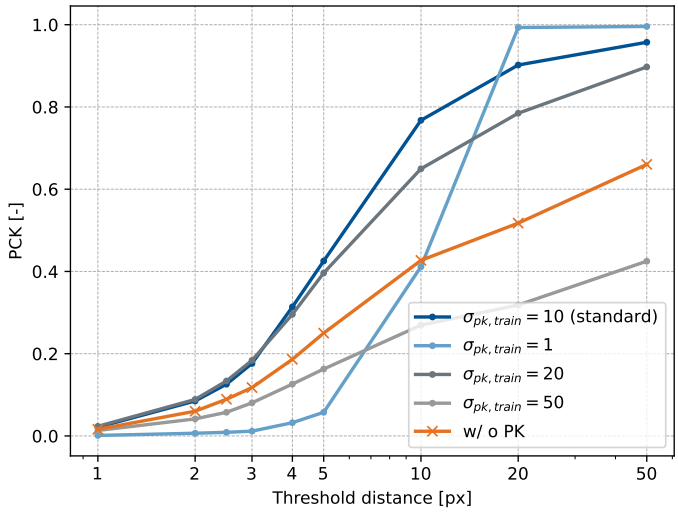
and on both metrics, as confirming our stated hypothesis, of incorporating prior kinematic knowledge and segmentation to introduce a better geometrical understanding to our network. This becomes especially observable the more uncertainties are inherent in a system, such as it is the case on the *neoDavid* arm. However, this system also highlights the limitations of our 2D keypoint detection approach as gains are marginal compared to the performance improvements realized on the other two datasets. Eventually, we conclude that the greatest effect of our prior kinematic approach is in case of reliable prior kinematic sources and a medium system complexity.

D. Ablation Studies

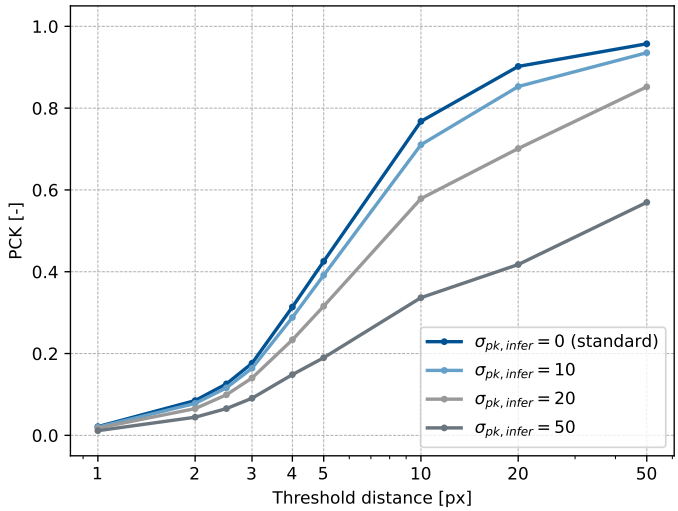
We conduct three ablation studies to evaluate our 2D keypoint detection. To this end, we run experiments to evaluate the impact of different prior knowledge perturbation strengths during training, $\sigma_{\text{train, pk}}$, and inference, $\sigma_{\text{infer, pk}}$, on our base network, i.e., our approach without the additional segmentation task, in order to have isolated findings without any potential confounder variable. Furthermore, we investigate whether our standard approach can recognize and measure uncertainty in our prior kinematic knowledge for variable $\sigma_{\text{smooth, train}}$ and $\sigma_{\text{train, pk}}$. Both these experiments are conducted on the *Jaco2* dataset. As a third ablation study, we examine the potential performance gain when we fine-tune our approach on real-world data of the *Panda* arm.

1) Impact of Prior Knowledge Perturbation Strength: In order to assess the impact of our choice of our prior knowledge perturbation $\sigma_{\text{pk, train}}$, we train our base approach with varying pixel strengths, ranging from one to 50 pixels. We hypothesize, the network learns to unconditionally trust the seen prior knowledge during training, in the case of a perturbation of just one pixel as this provides close to perfect and correct information. However, during inference, when the prior knowledge is not correct, this model does not achieve high accuracy detections beyond a *PCK* level of 20 pixels, as shown in Fig. 10a. Furthermore, we confirm our intuition of our prior kinematic knowledge approach as the more disturbed and thus imperfect it becomes, the less performance increase is attested. This empirical observation goes so far as the prior kinematic knowledge can even decrease accuracy performance compared to without any prior knowledge, as in Fig. 10a and Table I. We argue, that in such cases, the impact of our provided prior knowledge on the network is confusion rather than setting a region of interest to search for keypoints.

As stated, our approach to incorporate this prior kinematic



(a)



(b)

Fig. 10. Evaluation of our base 2D keypoint detection approach with varying σ_{pk} during training and inference on our real-world *Jaco2* data. The experiments with different $\sigma_{\text{pk, train}}$ reveal deteriorating accuracy with a increasing prior knowledge perturbation regime during training (a). Analogously, we observe performance decreases in case the prior knowledge perturbation during testing differs significantly from the training one (b).

knowledge is to provide an area to focus the keypoint prediction on and hence the network is supposed to balance this information with additionally learned clues from the RGB image $\boldsymbol{I}$. Thus, we analyze the performance of PK-ROKED during inference with varying inference perturbations when the network is trained with the best performing and our standard perturbation on the *Jaco2* of $\sigma_{\text{pk, train}} = 10$ pixels. The additional perturbation, $\sigma_{\text{pk, infer}}$, is applied on top of the already imperfect prior knowledge to make it even more imprecise, if not stated otherwise. We observe in Fig. 10b that our approach is balancing prior kinematic knowledge with additional learned information as hypothesized. We derive this by observing the only slight decrease in performance when the prior kinematic knowledge during inference is additionally disturbed with $\sigma_{\text{pk, infer}} = 10$ pixels. This gap would have been greater in case the network unconditionally trusts the prior kinematic knowledge – which it apparently does not. This is furthermore supported by the still robust accuracy in case of an even further and stronger inference perturbation.

Based on these two experiments, we conclude our approach of incorporating prior kinematic knowledge is balanced in such that it indicates a potential region of keypoints without our approach becoming purely reliant on this information modality. We observe that it seems not necessary to exactly set the parameter $\sigma_{\text{pk, train}}$ as the performance was still reasonable with other choices of the hyperparameter. Importantly, this parameter does not perfectly match $\sigma_{\text{pk, infer}}$ as we achieve good accuracy results even with further perturbations.

2) Measuring Prior Knowledge Uncertainty: Motivated by these results, we investigate whether our approach is capable to recognize to what extent it should rely on the provided prior kinematic knowledge. We hypothesize that our approach should be able to predict better keypoints with a well calibrated uncertainty in case of good prior knowledge, where the input is indicated to be more certain. In such a scenario, the network should be able to put more trust in this information and hence yield better results. To run such an experiment, we train a network with varying $\sigma_{\text{pk, train}}$ as well as different $\sigma_{\text{smooth, train}}$, each randomly chosen for a given training sample. To clarify, $\sigma_{\text{pk, train}}$ indicates how strong a given sample's prior knowledge is perturbed. In contrast to that $\sigma_{\text{smooth, train}}$ represents the area around a believed keypoint. Therefore, a wider area indicates less certainty and a narrow region is interpreted with higher certainty of the keypoint being at that location. For this experiment, we sample $\sigma_{\text{pk, train}} \in \{5, 10, 30\}$ and $\sigma_{\text{smooth, train}} \in \{2, 5, 15\}$ and train for 100 epochs as the network takes longer to convergence on the *Jaco2*. We evaluate the resulting model with two different variations of prior knowledge: first, the ground truth is the ideal prior knowledge, i.e., without any further perturbations applied s.t. $\sigma_{\text{pk, infer}} = 0 = \text{gt}$. Second, we sample the prior knowledge from the imprecise forward kinematics together with an additional standard deviation of $\sigma_{\text{pk, infer}} = 30$ pixels. For each of those two variants, we run the inference three times with varying σ_{smooth}, thus modify the indicated input uncertainty of the prior knowledge. Higher σ_{smooth} values result in greater keypoint areas of the input belief maps and thus indicate a greater uncertainty about the exact keypoint location compared to lower values. Along this line of reasoning, the network should predict less accurate keypoints in case of highly inaccurate prior knowledge which we falsely indicate to be certain. As shown in Fig. 11a, we observe the hypothesized performance. In case we feed our network perfect prior knowledge and indicate that this is certain, we observe the best performance, as depicted with $\sigma_{\text{smooth}} \in \{2, 5\}$. The accuracy decreases with increasing $\sigma_{\text{smooth}} = 15$, as we expected. This is further supported by the reasonably well calibrated *Precision* scores achieved in Fig. 11b, that closely reassembles a normal distribution.

In case of very strong additional perturbation on the prior kinematic knowledge, we observe the worst performance when we falsely indicate to our network that we are certain about it, i.e. $\sigma_{\text{pk, infer}} = 30$ together with $\sigma_{\text{smooth}} = 2$ and thus resulting in a strongly overconfident network. However, the performance clearly improves when we instead indicate uncertainty about

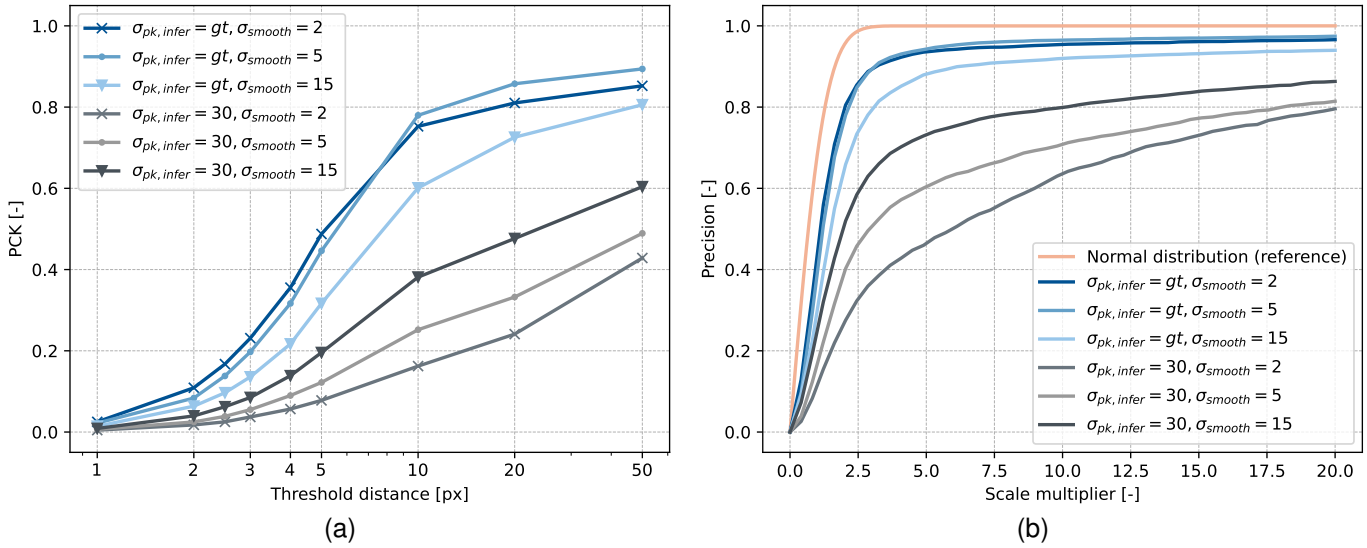


Fig. 11. Accuracy and *Precision* evaluation of our approach on the *Jaco2* dataset. The model is trained and evaluated with varying $\sigma_{\text{pk, train}}$ and $\sigma_{\text{smooth, train}}$. Thereby, $\sigma_{\text{pk, infer}} = \text{gt}$ indicates the ground truth which is fed to the network as ideal prior knowledge. Whereas $\sigma_{\text{pk, infer}} = 30$ implies an additional perturbation on the already imperfect prior knowledge obtained from the forward kinematics. We observe the best performance on both metrics when the model is run with perfect prior knowledge which we indicate to be certain about, i.e., $\sigma_{\text{pk, infer}} = \text{gt}$ and $\sigma_{\text{smooth}} \in \{2, 5\}$. The performance is worst in case of strong prior knowledge perturbation which is falsely believed to be certain. This experiment showcases that our approach is capable of recognizing the associated uncertainty of the received prior knowledge and to what extent it can rely on it for its keypoint predictions.

the strong perturbed prior knowledge. Hence, we deduce our approach can recognize whether and to what extent it should rely on the prior kinematic knowledge.

Furthermore, we highlight that when we evaluate this kind of model with our standard prior knowledge, which is already imperfect and a medium uncertainty of $\sigma_{\text{smooth}} = 5$, we surpass the accuracy of approaches without prior knowledge, as reported in Table II. Thus, we also conclude that knowing the exact parameters for $\sigma_{\text{pk, train}}$ and $\sigma_{\text{smooth, train}}$ is not required in order to benefit from the performance gains of incorporating prior kinematic knowledge.

TABLE II
ACCURACY OF PK-ROKED TRAINED WITH VARYING $\sigma_{\text{SMOOTH, TRAIN}}$ AND $\sigma_{\text{PK, TRAIN}}$.

Method	*Jaco2* *PCK* (@px) ↑				
	1	3	5	10	50
Ours w/ o PK + seg.	**0.016**	0.118	0.25	0.427	0.66
Our w/ var. hyper.	**0.016**	**0.147**	**0.336**	**0.678**	**0.877**

3) Fine-tuning on Real-world Data: Furthermore, we analyze the upper limit of our approach in case it is further fine-tuned on a non-overlapping subset of the real-world test dataset. We argue that this real-world fine-tuning of PK-ROKED has direct practical application, in cases where our approach is used to initialize a tracker as shown in Section VIII. Thereby, we collect data and gather pseudo labels through deploying the tracking as ground truth. We successfully applied and tested this routine on *neoDavid* and boosted the accuracy from 0.277 to 0.48 at the *PCK*@5 pixels through fine-tuning on additional 800 images for 40 epochs. Similarly, we report such a fine-tuning experiment on a non-overlapping subset of the real-world *Panda* dataset. We conduct two such runs, one trained on ca. 160 images, which represents only 0.05% of the dataset and another model fine-tuned on 1% of a non-overlapping subset. Both train for additional 20 epochs, and we decrease the learning rate to $lr = 1.5\text{e-}5$. In Table III, we observe strong performance improvements across all *PCK* levels. We also notice that the fine-tuned models achieve the highest accuracy on the lower *PCK* values and are on par with CtRNet on the higher ones, or even surpass it, as reported in Table I. Thus, incorporating real-world data opens up as a promising future research path.

TABLE III
PCK FINE-TUNING EVALUATION ON THE *Panda* DATASET.

Method	Fine-tuning on *Panda* *PCK* (@px) ↑				
	1	3	5	10	50
Ours (synth.)	0.068	0.491	0.793	0.944	0.992
Ours (0.05%)	*0.195*	*0.706*	*0.879*	*0.972*	**0.995**
Ours (1%)	**0.416**	**0.857**	**0.939**	**0.978**	*0.989*

VI. EVALUATING THE PNP-POSE ON *Panda*

This section presents results on the 6D pose estimation on the *Panda* dataset based on the standard PnP algorithm using predicted 2D keypoints, analogously to DREAM. We emphasize that deriving the 6D pose by means of a PnP algorithm comes with disadvantages, as outlined in Section III-B and thus is not meant to be our primary method to estimate the 6D robot pose. Nevertheless, it is well suited to compare PK-ROKED with the state-of-the-art. This evaluation allows us to compare our approach to additional methods such as CtRNet-X [36] and Real-Time Holistic Robot Pose Estimation with Unknown States [26], which we refer to as HPE.

A. Metric for 6D PnP-Evaluation

The 6D pose on the *Panda* dataset is commonly evaluated by the *average distance (ADD)* [6]. The *ADD* measures the accuracy between 3D points and their transformed counterparts, mapped by the derived 6D camera-to-robot pose. Analogously to the *PCK* metric, the percentage of *ADD* is evaluated at various thresholds. This is reported as the *area-under-the-curve (AUC)* and the mean error. For the former, higher values show better accuracy and for the latter one, smaller average errors are desirable.

TABLE IV
6D ACCURACY EVALUATION AS MEASURED BY *ADD* AND THE CORRESPONDING *AUC* AND MEAN (M).

Method	*Panda*	
	Mean (m) ↓	*AUC* ↑
Ours (synth.)	0.034	75.46
Ours (0.05%)	0.017	*88.12*
Ours (1%)	*0.015*	**89.29**
DREAM	0.025	69.1
CtRNet	0.021	85.29
CtRNet-X	**0.014**	*86.23*
RoboPEPP	[-]	77.5
HPE	0.025	75.20

B. 6D Pose Evaluation Results

In Table IV, we observe that our only synthetically trained model already performs on par with the current state-of-the-art and is just outperformed by CtRNet and its follow-up work. This is analogously to our findings on the 2D evaluation and is also likely to be due to the real-world self-supervised training deployed by the CtRNet approaches. However, we achieve comparable results to HPE and RoboPEPP, which both also perform self-supervised training on real-world data. Their main respective difference to the CtRNet approaches is that the latter assumes to have access to the joint values, whereas the former two approaches estimate these as well. Additionally, in case of our fine-tuned version, we even achieve the best performance on the *AUC* with just a slightly inferior mean error to CtRNet-X.

VII. Probabilistic Pose Estimation Evaluation

So far, we have evaluated the performance of the standalone keypoint detection as it provides the two-dimensional keypoint positions on the camera image. However, in the end most robotic applications require spatial information as input, either as 3D points or 6D poses. To achieve this, we use the 2D information on the keypoint location and combine it with the 6D forward kinematics to obtain a valid and corrected 6D pose of that keypoint. Note that – for comparability of the experiment results – we treat each keypoint independently in this experiment section. Furthermore, we only use a single shot fusion approach for comparability, as continuous pose estimation methods can be very system specific and their approaches highly depend on factors like measurement rate of the sensors or observability of kinematic errors. More thorough fusion approaches such as continuous pose estimation and rigid body estimations using multiple keypoints is out of scope and left to future work.

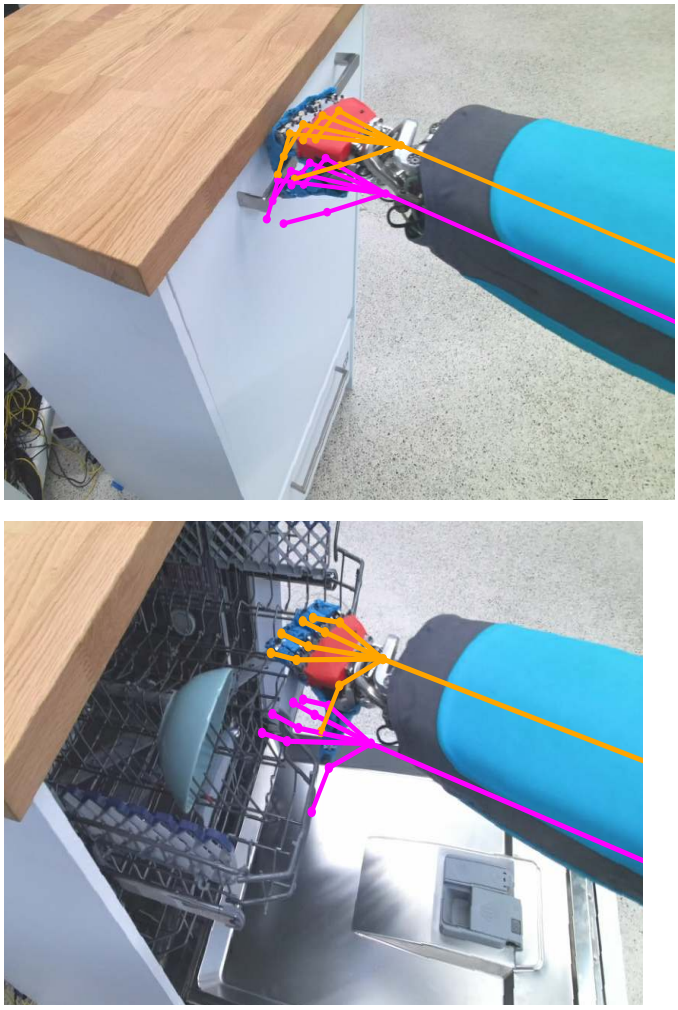


Fig. 12. Evaluation of our fusion results on the *neoDavid* data, while the robot empties a dishwasher. The pose estimation of our approach is highlighted with the orange skeleton. Whereas, the forward kinematics are shown in magenta. As one observes, our approach is able to significantly improve the pose estimation.

Recall that our fusion approach is detailed in Section III-B2. The 2D output of PK-ROKED is our input for the correction step and the probabilistic kinematics [8] are the input for the prediction step.

We evaluate the fusion considering two experiments on our two different robotic datasets – *neoDavid* and *LRU2* with its *Jaco2* arm. We compare the resulting keypoint poses to the ground truth. Similarly, we compare the uncorrected robot kinematics to the ground truth and subsequently show the improvement in pose estimation of our fused keypoint detection compared to the pure forward kinematics.

A. Metric for Pose Evaluation

Both experiments feature external ground truth measures for the true keypoint pose. Our evaluation metric is therefore the *positional error* between a measured pose and the ground truth pose.

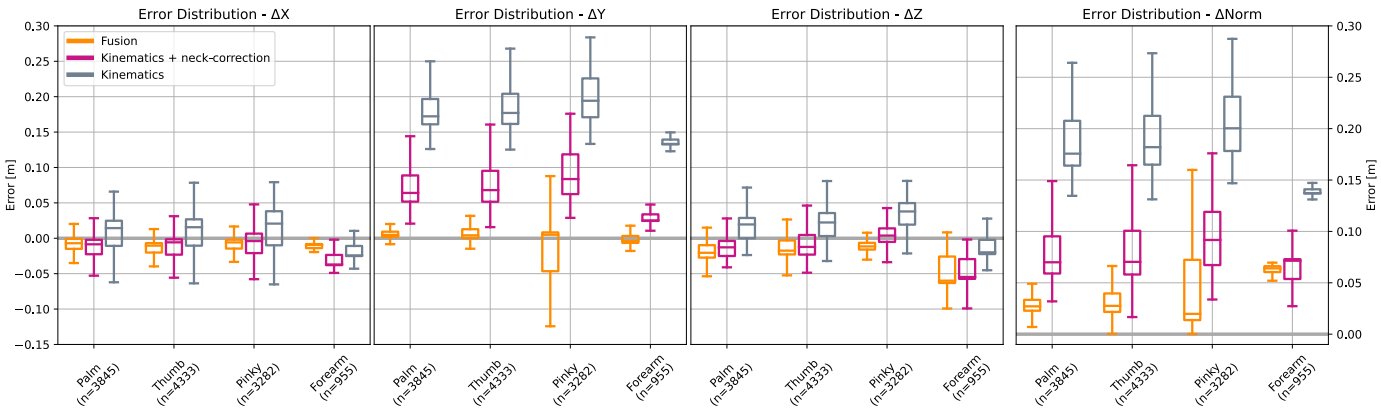


Fig. 13. Experiments on *neoDavid*: we evaluate our pose estimation while *neoDavid* empties the dishwasher, using our M3T tracking as ground truth. The accuracy of the pose estimation is shown in all spatial dimensions and in terms of Euclidean norm as difference between the obtained pose and the ground truth pose.

B. Pose Fusion Results on neoDavid

First, we evaluate the pose estimation of the keypoints on our robot *neoDavid*.

To enable the sensor fusion, the probabilistic kinematics need to be parameterized. We assign the highest uncertainty to the neck joint of *neoDavid*, with a standard deviation of $10\,\text{mm}$ and $0.1\,\text{rad}$, for each translational and rotational component, respectively. The arm joints feature a joint-error along the principal rotation axis, which we model as $0.01\,\text{rad}$ and the nonlinear wrist has an assigned uncertainty of $0.1\,\text{rad}$ standard deviation in both roll and pitch. Finally, each finger element would feature a high uncertainty as well, especially as these joint values are only measured indirectly, but as we only consider keypoints at the finger base, this is irrelevant for the current evaluation.

We run PK-ROKED and obtain the 2D position and associated covariance for four selected keypoints that are placed on or close to the robotic hand: one is located at the end of the palm; two are at the root of thumb and little finger, respectively; and one is located on the robot's forearm. Figure 1(bottom left) shows a PK-ROKED detection of the first three listed keypoints. Note that we train the keypoint detection model on a fifth keypoint, the right elbow, for our real-world applications and daily use. However, we neglect this keypoint in the 6D evaluation as it is barely visible in the recorded dataset.

The fusion step is used to compute the improved keypoint pose which is then used to correct the overall robot kinematics. The resulting – corrected – skeleton is shown in Fig. 1(bottom right). We consider the robot's forward kinematics as reference. Note that *neoDavid*'s kinematics offer peculiarities: we already employ neck-pose-correction algorithms [12] to get an improved relative transformation between the head and the robot's torso, compared to the nominal forward kinematics which approximates the neck as two revolute joints. The kinematics with the neck-pose-correction are referred to as *kinematics-and-neck* and are visualized in Fig. 1(bottom) and Fig. 12 along our fusion results. We use both kinematics sources in our evaluation.

For the accuracy evaluation, we compare both kinematics sources and our fusion with the ground truth and apply the position error as evaluation metric. The results are shown in Fig. 13, where the directional errors are w.r.t. the camera frame. Generally, our pose fusion of PK-ROKED and the probabilistic kinematics provides the best accuracy for the robot arm pose. The kinematics including the neck correction outperform the simplified kinematics.

Notably, the error of the kinematics mostly affects the y direction. This is due to the fact that the neck error is locally constant. During the dishwasher demo, the neck only moves slightly around an initial position that favors the errors in one direction over the other.

Regarding the individual keypoints, the fusion results for palm and thumb generally have very good results with few outliers. Contrary to that, the pose estimation using the little finger keypoint performs well in many cases, but has both a significant higher outlier rate than the previously mentioned keypoints and multiple errors in the $5\,\text{cm}$ regime. The $5\,\text{cm}$ error is explained by an observed detection ambiguity between the base of the index finger and the base of the little finger, causing the network to confuse both and assigning a false positive little finger detection to the index finger on multiple occasions. We also attribute the higher outlier rate for the little finger to the same ambiguity, as the network confuses these two finger keypoints and their location on the hand. Finally, the forearm performs reasonably well, however it has to be stated that this keypoint is rarely visible, thus providing only limited data points for comparison.

C. Pose Fusion Results on LRU2 with Jaco2

Finally, we evaluate our algorithm on the *LRU2* system. We model the kinematic errors similar to the *neoDavid* case. The bending of the base-plate is modeled with an uncertainty of $[0.02, 0.1, 0.001]\,\text{rad}$ standard deviation, with the pitch experiencing the highest uncertainty due to the gravity induced bending of the plate. Each joint gets assigned $0.02\,\text{rad}$ and the pan-tilt camera mount is set to $0.01\,\text{rad}$ for the pan and tilt directions.

The resulting pose estimate is visualized in Fig. 15 for two example configurations. There, we illustrate the robot skeleton

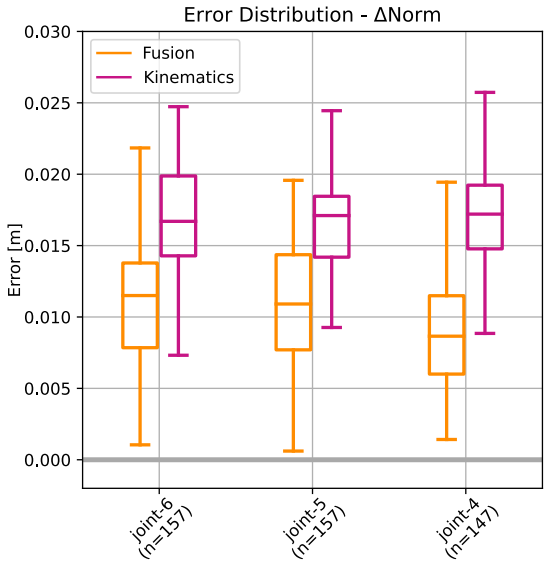


Fig. 14. The accuracy of the pose estimation in terms of Euclidean norm, using the VICON tracking as ground truth.

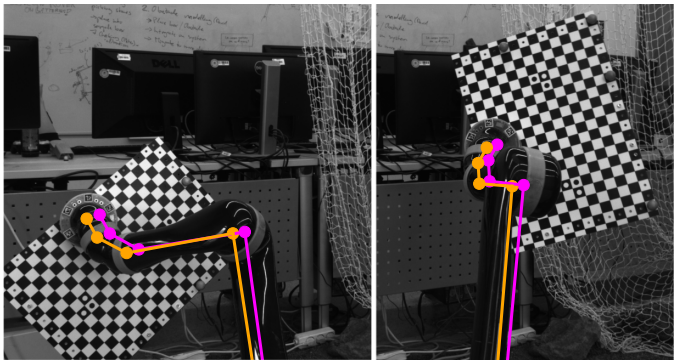


Fig. 15. The *Jaco2* experiments: a calibration plate with integrated VICON markers is attached to the end-effector of the arm. The arm moves to multiple configurations while being observed by the *LRU2*'s camera. The pose estimation result is shown as the orange skeleton compared to the inaccurate forward kinematics (magenta skeleton). Note that we convert the RGB images to grayscale for the visualization purposes.

from the forward kinematics (magenta), which clearly shows an offset to the true arm pose, however less severe as compared to the kinematics errors at the *neoDavid* robot. The figure also shows the skeleton of the corrected robot arm pose, for which we use the fusion result of the end-effector keypoint and draw the skeleton using this result as relative reference.

We illustrate the accuracy of our pose estimation in Fig. 14. We limit the evaluation only to the static phases of the experiment: even though the camera images and the VICON ground truth are accurately synchronized in time, it turns out that there is a non-constant time offset between the joint readings and all other data. Our focus on static configurations allows us to rule out an incorrect computation of the residual estimation error due to a timing-mismatch.

The evaluation shows that our pose estimation allows correcting the robot's forward kinematics. However, it is still susceptible to noise and outliers as the upper quartile of the pose error lies in the range of the kinematics error. This motivates future work: either combine the measurements of several keypoints using the rigid body assumption and/or combine the observed keypoints over multiple images in a smoothing approach for outlier reduction.

VIII. Robotic Application: Tracker Initialization and Tracker Monitoring

As outlined in Section I, one approach to mitigate inaccurate pose estimates from forward kinematics are continuous tracking methods, such as M3T [5]. These approaches however require an initialization of the tracking object. On *neoDavid*, the arm tracking is currently initialized by starting the tracker in a pre-defined arm configuration. The obvious disadvantage is the static initialization itself, as it requires returning to the start configuration each time the tracker is lost. Hence, a dynamic initialization is desired which in turn would allow for online re-initialization and thus a smoother recovery from lost tracking situations.

To this end, we configure the M3T tracking pipeline implemented on *neoDavid* to take as initialization input the pose output of our approach, i.e., the fused 6D poses. We run this experiment on various images and poses from our *neoDavid* dataset. Additionally, we qualitatively compare our initialization of the tracker with those solely based on the forward kinematics.

We judge the initialization poses, represented as pink arm overlays, by how much they visually match with the actual arm in a given image. As we observe in Fig. 16, we obtain significantly better initialization poses in case we feed our poses to the tracker versus the forward kinematics only. This has a significant impact on the following tracking success and therefore the continuous and consecutive pose estimation. A tracker, such as M3T, is able to refine local pose correction through optimization – as long as they are of just minor magnitude. It is not, however, able to counter-act cases when the initialization is too much off from the actual target, as in case of the experiment with the forward kinematics only.

To conclude, we deem the experiment to initialize a tracker with our PK-ROKED approach as successful. Thereby, it highlights further the versatility and applicability of our approach. Eventually, this demonstrates a promising robotics application with real-world impact as it allows initializing a tracker from various positions. Hence, it increases the potential application space as well as autonomy of a robotic system, as demonstrated on the DLR's *neoDavid*. In the future we envision a parallel execution of the tracker and PK-ROKED, where the former module provides fast and accurate tracking results and the latter continuously supervises for a potential loss-of-tracking and re-initialization of the tracker if necessary.

IX. Conclusion

In this work, we developed the Prior Knowledge Robot Keypoint Detection (PK-ROKED) to correct the arm pose of robotic systems that suffer from inaccurate forward kinematics. To this end, our method first detects 2D keypoints on a robot arm given a single RGB image and imperfect prior kinematic knowledge. The latter is derived from the inaccurate

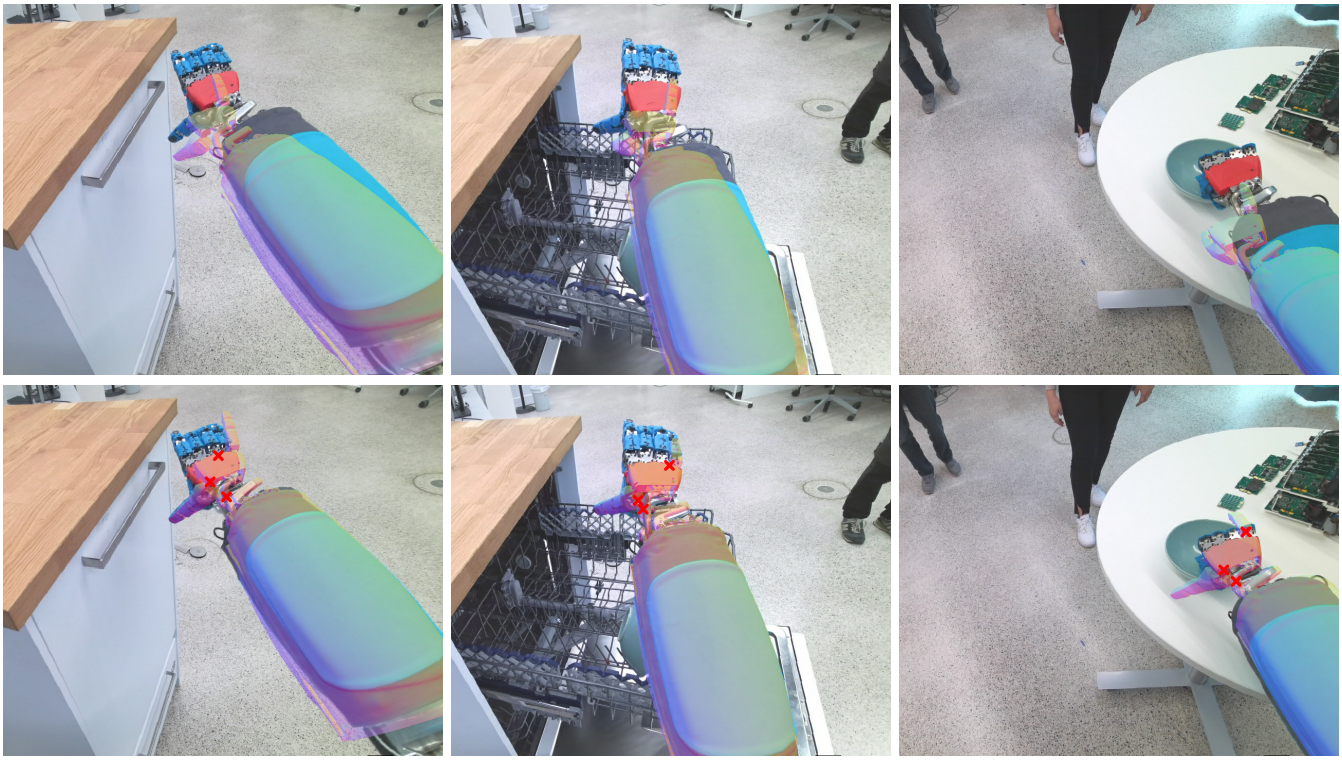


Fig. 16. Qualitative comparison to initialize the M3T arm tracker through the pose based on our approach versus the forward kinematics only. The pink overlay is the initialized position based on which the tracker would start refining. The upper row presents the initialization as obtained through the forward kinematics. In the lower row we see the initialization with poses derived by our approach. Red markers indicate in this image the input poses to the M3T tracker.

forward kinematics, which is imprecise but stays within a bounded error. Second, based on the resulting 2D predictions and their corresponding uncertainties, we lift the 2D keypoints to actual 6D poses by fusing them with the inaccurate forward kinematics, requiring only one successfully detected keypoint.

We demonstrated the performance of PK-ROKED on two different robots in addition to a benchmark dataset. By reporting quantitative results on these datasets, we showed the advantage of our approach as we outperform other 2D keypoint detections methods. Thereby, we confirm our hypothesis that prior kinematic knowledge improves keypoint detections, as shown by the measured accuracy gains. The generalizibility of our approach is highlighted twofold: first, by incorporating our prior kinematic knowledge concept into another method DREAM, and thus improving the respective performance and second, by deploying our approach on the three different systems, which vary in complexity.

In addition to the 2D performance, we evaluated our performance on the ability to correct an erroneous pose estimation. Therefore, we deployed the approach on our systems and reported pose estimation improvements on both. An important aspect for this is our contribution of computing uncertainties for our keypoint predictions. By this, we were able to apply a probabilistic fusion framework, which works with only one keypoint detection - a distinction to other methods.

However, we also encountered some limitations of our approach, which we are eager to tackle and improve upon in the future. A limiting factor for our performance gain is the quality of aforementioned prior knowledge. As demonstrated on the *neoDavid* robot and further ablated on the *Jaco2*, the more uncertain and hence off the prior kinematic becomes, the less valuable it is regarded. We conclude that the performance gains especially w.r.t. accuracy are dependent on the prior knowledge quality and to what extent our approximation of modeling it through Gaussian perturbations is reflecting reality. Thus, we plan to improve the modeling of the prior kinematic knowledge by learning its actual distribution for a given robot and corresponding kinematic. As future work to improve the fusion step, we consider advancing it towards a continuous pose estimation method that uses multiple keypoints.

Furthermore, we demonstrated the applicability of our method by integrating it with a continuous tracker on a real robotic system. Here, we showed that with our approach the initialization of a robot arm tracking – one of the main pain points of those approaches – can be overcome.

We sincerely hope, this work sparks some imitation in the robotic computer vision community to consider incorporating such valuable and available information as forward kinematics – even if not completely accurate.

ACKNOWLEDGMENTS

This work is supported by the European Union's Horizon Europe research and innovation framework under grant agreement No. 101070136, project IntelliMan.

REFERENCES

[1] M. Sundermeyer, Z.-C. Marton, M. Durner *et al.*, "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images," in *European Conference on Computer Vision*, 2018, pp. 699–715.

[2] Y. Labbé, J. Carpentier, M. Aubry *et al.*, "CosyPose: Consistent multi-view multi-object 6D pose estimation," in *European Conference on Computer Vision*, Aug. 2020.

[3] C. Della Santina, M. G. Catalano, and A. Bicchi, "Soft Robots," in *Encyclopedia of Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2020, pp. 1–15.

[4] C. Nissler, M. Durner, Z.-C. Marton *et al.*, "Simultaneous Calibration and Mapping," in *Experimental Robotics*, Nov. 2018.

[5] M. Stoiber, M. Sundermeyer, W. Boerdijk *et al.*, "A Multi-body Tracking Framework - From Rigid Objects to Kinematic Structures," *arXiv*, Feb. 2023.

[6] T. E. Lee, J. Tremblay, T. To *et al.*, "Camera-to-Robot Pose Estimation from a Single Image," in *IEEE International Conference on Robotics and Automation*, May 2020, pp. 9426–9432.

[7] L. Meyer, L. Klüpfel, M. Durner *et al.*, "Robust Probabilistic Robot Arm Keypoint Detection Exploiting Kinematic Knowledge," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on Probabilistic Robotics in the Age of Deep Learning*, 2022.

[8] L. Meyer, K. H. Strobl, and R. Triebel, "The Probabilistic Robot Kinematics Model and its Application to Sensor Fusion," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2022, pp. 3263–3270.

[9] S. Wolf, C. Hofmann, T. Bahls *et al.*, "Modularity in Humanoid Robot Design for Flexibility in System Structure and Application," in *IEEE International Conference on Humanoid Robots*, Dec. 2023, pp. 1–7.

[10] M. J. Schuster, S. G. Brunner, K. Bussmann *et al.*, "Towards Autonomous Planetary Exploration," *Journal of Intelligent & Robotic Systems*, vol. 93, no. 3, pp. 461–494, Nov. 2017.

[11] B. Vanderborght, A. Albu-Schaeffer, A. Bicchi *et al.*, "Variable impedance actuators: A review," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1601–1614, Dec. 2013.

[12] A. Raffin, B. Deutschmann, and F. Stulp, "Fault-Tolerant Six-DoF Pose Estimation for Tendon-Driven Continuum Mechanisms," *Frontiers in Robotics and AI*, vol. 8, Apr. 2021.

[13] M. Grebenstein, A. Albu-Schäffer, T. Bahls *et al.*, "The DLR hand arm system," in *IEEE International Conference on Robotics and Automation*, May 2011, pp. 3175–3182.

[14] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *IEEE International Conference on Robotics and Automation*, May 2011, pp. 3400–3407.

[15] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas *et al.*, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, Jun. 2014.

[16] K. H. Strobl and G. Hirzinger, "Optimal Hand-Eye Calibration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 4647–4653.

[17] O. Birbach, U. Frese, and B. Bäuml, "Rapid calibration of a multi-sensorial humanoid's upper body: An automatic and self-contained approach," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 420–436, Apr. 2015.

[18] M. Klingensmith, T. Galluzzo, C. M. Dellin *et al.*, "Closed-loop Servoing using Real-time Markerless Arm Tracking," in *IEEE International Conference on Robotics and Automation*, 2013.

[19] T. Schmidt, K. Hertkorn, R. Newcombe *et al.*, "Depth-based tracking with physical constraints for robot manipulation," in *IEEE International Conference on Robotics and Automation*, May 2015, pp. 119–126.

[20] T. Schmidt, R. Newcombe, and D. Fox, "DART: Dense Articulated Real-Time Tracking," *Robotics: Science and Systems*, vol. 2, pp. 1–9, Jul. 2014.

[21] C. Garcia Cifuentes, J. Issac, M. Wüthrich *et al.*, "Probabilistic Articulated Real-Time Tracking for Robot Manipulation," *IEEE Robotics and Automation Letters*, vol. 2, pp. 577–584, Apr. 2017.

[22] J. Bohg, J. Romero, A. Herzog *et al.*, "Robot arm pose estimation through pixel-wise part classification," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 3143–3150.

[23] F. Widmaier, D. Kappler, S. Schaal *et al.*, "Robot arm pose estimation by pixel-wise regression of joint angles," in *IEEE International Conference on Robotics and Automation*, May 2016, pp. 616–623.

[24] Y. Labbe, J. Carpentier, M. Aubry *et al.*, "Single-View Robot Pose and Joint Angle Estimation via Render & Compare," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1654–1663.

[25] S. Ban, J. Fan, X. Ma *et al.*, "Real-time Holistic Robot Pose Estimation with Unknown States," in *European Conference on Computer Vision*, Jul. 2024.

[26] R. G. Goswami, P. Krishnamurthy, Y. LeCun *et al.*, "RoboPEPP: Vision-Based Robot Pose and Joint Angle Estimation through Embedding Predictive Pre-Training." arXiv, Nov. 2024.

[27] L. Chen, Y. Qin, X. Zhou *et al.*, "EasyHeC: Accurate and Automatic Hand-eye Calibration via Differentiable Rendering and Space Exploration," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7234–7241, Nov. 2023.

[28] Z. Hong, K. Zheng, and L. Chen, "EasyHeC++: Fully Automatic Hand-Eye Calibration with Pretrained Image Models," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2024.

[29] J. Lambrecht, "Robust Few-Shot Pose Estimation of Articulated Robots using Monocular Cameras and Deep-Learning-based Keypoint Detection," in *IEEE International Conference on Robot Intelligence Technology and Applications*, Nov. 2019, pp. 136–141.

[30] J. Lambrecht and L. Kästner, "Towards the Usage of Synthetic Data for Marker-Less Pose Estimation of Articulated Robots in RGB Images," in *International Conference on Advanced Robotics*, Dec. 2019, pp. 240–247.

[31] Y. Zuo, W. Qiu, L. Xie *et al.*, "CRAVES: Controlling Robotic Arm With a Vision-Based Economic System," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4214–4223.

[32] J. Lu, F. Richter, and M. C. Yip, "Pose Estimation for Robot Manipulators via Keypoint Optimization and Sim-to-Real Transfer," *IEEE Robotics and Automation Letters*, vol. 7, pp. 4622–4629, Apr. 2022.

[33] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An Accurate O(n) Solution to the PnP Problem," *International Journal of Computer Vision*, vol. 81, no. 2, pp. 155–166, Feb. 2009.

[34] J. Lambrecht, P. Grosenick, and M. Meusel, "Optimizing Keypoint-based Single-Shot Camera-to-Robot Pose Estimation through Shape Segmentation," in *IEEE International Conference on Robotics and Automation*, May 2021, pp. 13 843–13 849.

[35] J. Lu, F. Richter, and M. C. Yip, "Markerless Camera-to-Robot Pose Estimation via Self-supervised Sim-to-Real Transfer," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Mar. 2023.

[36] J. Lu, Z. Liang, T. Xie *et al.*, "CtRNet-X: Camera-to-Robot Pose Estimation in Real-world Conditions Using a Single Camera." arXiv, Sep. 2024.

[37] Y. Tian, J. Zhang, Z. Yin *et al.*, "Robot Structure Prior Guided Temporal Attention for Camera-to-Robot Pose Estimation from Image Sequence," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2023, pp. 8917–8926.

[38] Y. Tian, J. Zhang, G. Huang *et al.*, "RoboKeyGen: Robot Pose and Joint Angles Estimation via Diffusion-based 3D Keypoint Generation," in *IEEE International Conference on Robotics and Automation*, 2024.

[39] K. He, X. Zhang, S. Ren *et al.*, "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2016, pp. 770–778.

[40] O. Russakovsky, J. Deng, H. Su *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Jan. 2015.

[41] A. Kendall and Y. Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" in *Neural Information Processing Systems*, vol. 30, 2017.

[42] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *International Conference on Machine Learning*, Jun. 2016, pp. 1050–1059.

[43] N. Srivastava, G. Hinton, A. Krizhevsky *et al.*, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[44] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding," *arXiv*, Oct. 2016.

[45] J. Mukhoti and Y. Gal, "Evaluating Bayesian Deep Learning Methods for Semantic Segmentation," *arXiv*, Mar. 2019.

[46] J. Flusser, T. Suk, and B. Zitová, *Moments and Moment Invariants in Pattern Recognition*, 1st ed. Wiley, Oct. 2009.

[47] L. Ferraz, X. Binefa, and F. Moreno-Noguer, "Leveraging feature uncertainty in the pnp problem," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[48] G. Bourmaud, R. Mégret, A. Giremus *et al.*, "Discrete extended kalman filter on lie groups," in *European Signal Processing Conference*, 2013, pp. 1–5.

[49] J. Solà, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," *arXiv*, 2018.
[50] K. H. Strobl, W. Sepp, S. Fuchs *et al.* (2024) DLR CalDe and DLR CalLab. Institute of Robotics and Mechatronics, German Aerospace Center (DLR). Oberpfaffenhofen, Germany.
[51] A. Wedler, M. G. Müller, M. Schuster *et al.*, "Finally! insights into the arches lunar planetary exploration analogue campaign on etna in summer 2022," in *International Astronautical Congress*, Sep. 2022.
[52] M. Stoiber, M. Sundermeyer, and R. Triebel, "Iterative corresponding geometry: Fusing region and depth for highly efficient 3d tracking of textureless objects," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2022, pp. 6845–6855.
[53] M. Denninger, D. Winkelbauer, M. Sundermeyer *et al.*, "BlenderProc2: A Procedural Pipeline for Photorealistic Rendering," *Journal of Open Source Software*, vol. 8, no. 82, p. 4901, Feb. 2023.
[54] J. Tobin, R. Fong, A. Ray *et al.*, "Domain randomization for transferring deep neural networks from simulation to the real world," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2017, pp. 23–30.
[55] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," in *International Conference on Learning Representations*, Jan. 2019.
[56] Y. Yang and D. Ramanan, "Articulated Human Detection with Flexible Mixtures of Parts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2878–2890, Dec. 2013.

Leonard Klüpfel is currently a Ph.D. student at KIT and a research scientist in the department for Perception and Cognition at the DLR Institute of Robotics and Mechatronics. He received his B.Sc. in Industrial Engineering from the Friedrich-Alexander University of Erlangen-Nürnberg and his M.Sc. in Robotics, Cognition, Intelligence from TUM. His research focuses on computer vision for semantic scene understanding in the context of robotic manipulation with a special interest in detection and tracking of articulated objects with physical considerations and inspiration.

Lukas Burkhard is a researcher in the department of Perception and Cognition at the DLR Institute of Robotics and Mechatronics, since 2018. He works in the field of planetary robotics and humanoid robots. His focus is on sensor fusion, perception for robust manipulation, and additionally rover navigation algorithms. Lukas received his Master's and Bachelor's degree in Mechanical Engineering and Engineering Science from TUM.

Anne Elisabeth Reichert is a computer vision researcher in the department of Perception and Cognition at the Institute of Robotics and Mechatronics, DLR. Her research area is semantic aware robotic manipulation with special interest in 6D pose tracking of (articulated) objects based on visual input and kinematic considerations. Anne received her Master's degree in Robotics, Cognition, Intelligence from TUM and a Bachelor's degree in Computer Science from Baden-Württemberg Cooperative State University.

Maximilian Durner is a research fellow at the Institute of Robotics and Mechatronics, DLR since 2016 and a Ph.D. student at TUM. Before, he studied Electrical Engineering at TUM, partially studying at the Politecnico di Torino, Italy, and at the Universidad Nacional de Bogota, Colombia. Currently, he leads a research group on semantic scene analysis in the Perception and Cognition Department at DLR, which is working on robotic perception providing semantic information to interact with the surroundings. This includes known, unseen, and unknown object detection, pose estimation, and tracking. In this context, Maximilian focuses on robust and continuously adapting object-centric perception for manipulation.

Rudolph Triebel received his PhD in 2007 from the University of Freiburg in Germany. The title of his PhD thesis is "Three-dimensional Perception for Mobile Robots". From 2007 to 2011, he was a post-doctoral researcher at ETH Zurich, where he worked on machine learning algorithms for robot perception within several EU-funded projects. Then, from 2011 to 2013 he worked in the Mobile Robotics Group at the University of Oxford, where he developed unsupervised and online learning techniques for detection and classification applications in mobile robotics and autonomous driving. From 2013 to 2023, Rudolph worked as a lecturer at TUM, where he teached master level courses in the area of Machine Learning for Computer Vision. In 2015, he was appointed as leader of the Department of Perception and Cognition at the Robotics Institute of DLR, and in 2023 he was appointed as a university professor at KIT in "Intelligent Robot Perception".

APPENDIX

DATA AUGMENTATIONS

Table V shows the applied data augmentations during the training phase of the keypoint detection network. The listed techniques are used on all three synthetic datasets as augmentations to further bridge the difference between synthetic and real-world images of the respective robot arms.

TABLE V
DATA AUGMENTATIONS FOR INPUT IMAGES. GAUSSIAN NOISE IS APPLIED TO EACH INPUT IMAGE. ADDITIONALLY, ONE AUGMENTATION FROM EACH "ONE OF" BLOCK IS CHOSEN, ACCORDING TO THE PROBABILITY VALUES.

	Augmentation	Probability
	Gaussian noise	$p = 1.0$
One of	RGB channel shuffle	$p = 0.23$
	Random brightness contrast	$p = 0.31$
	RGB shift	$p = 0.27$
	Hue, saturation and value	$p = 0.19$
One of	Horizontal flip	$p = 0.28$
	Vertical flip	$p = 0.28$
	Coarse dropout	$p = 0.44$