Probabilistic Articulated Real-Time Tracking for Robot Manipulation

Cristina Garcia Cifuentes[1], Jan Issac[1], Manuel Wüthrich[1], Stefan Schaal[1,2] and Jeannette Bohg[1]

***Abstract*— We propose a probabilistic filtering method which fuses joint measurements with depth images to yield a precise, real-time estimate of the end-effector pose in the camera frame. This avoids the need for frame transformations when using it in combination with visual object tracking methods.**

Precision is achieved by modeling and correcting biases in the joint measurements as well as inaccuracies in the robot model, such as poor extrinsic camera calibration. We make our method computationally efficient through a principled combination of Kalman filtering of the joint measurements and asynchronous depth-image updates based on the *Coordinate Particle Filter*.

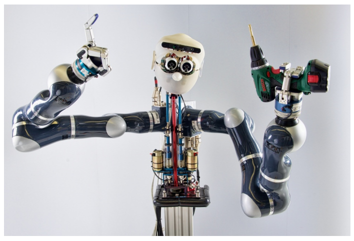
We quantitatively evaluate our approach on a dataset recorded from a real robotic platform, annotated with ground truth from a motion capture system. We show that our method is robust and accurate even under challenging conditions such as fast motion, significant and long-term occlusions, and time-varying biases. We release the dataset along with open-source code of our method to allow quantitative comparison with alternative approaches.

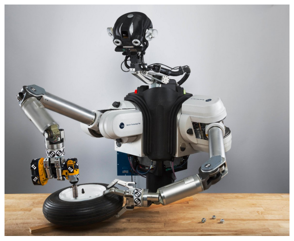
I. INTRODUCTION

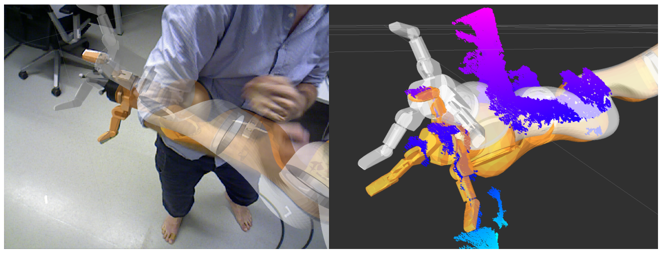
Autonomous grasping and manipulation remain a frontier in robotics research, especially in complex scenarios which are characterized by unstructured, dynamic environments. Under such conditions, it is impossible to accurately predict all consequences of an action far into the future. Therefore, open-loop execution of offline-planned manipulation actions is very likely to fail in such scenarios.

A key ingredient for accurate manipulation is to *continuously* and *precisely* estimate the configuration of the robot's manipulator and the target objects. This configuration can be expressed as the 6-degree-of-freedom (DoF) poses of all objects of interest in a common frame of reference. The focus of this paper is on the estimation of the end-effector pose with respect to the camera frame. We refer to this problem as *robot tracking*. The most common approach to estimating the end-effector pose is to apply forward kinematics using joint angle measurements. However, errors in the joint measurements are common, due to sensor drift or bias, and complex mechanical effects like cable stretch. Inaccuracies in the kinematic model are quite common as well, because the locations of different parts of the robot with respect to each other might not be perfectly known. Even slight errors in either of these two may lead to large errors in the end-effector pose. On the other hand, depth cameras can be used to accurately estimate the end-effector pose, but images are typically received at a lower rate, often with significant delay. Additionally, the computation required to obtain an estimate from a depth image is typically large, which further increases the delay until the estimate is available.

In order to achieve precise, real-time robot tracking, we present a method that combines joint measurements with depth images from a camera mounted on the robot, so as to get the advantages from both: accurate, up-to-date estimates at a high rate. We formulate the problem in the framework of recursive Bayesian estimation, enabling principled, online fusion of the two sources of information. We account for the


(a) Apollo


(b) ARM


(c) Sequence from Apollo with large simulated bias and strong occlusions

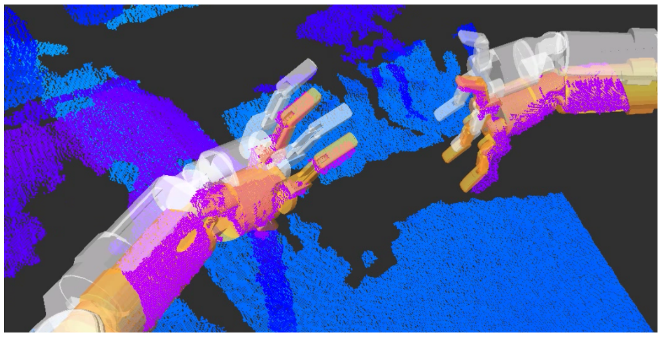

(d) Sequence from ARM with moderate real biases

Fig. 1. Our method produces real-time, accurate end-effector poses in the camera frame, despite of joint encoder biases and/or poor extrinsic calibration of the camera to the robot. (a, b) The two robotic platforms used for data recording and evaluation. (c, d) We can see that for both robots our estimate (in orange) aligns well with the visual information. This enables precise interaction with objects tracked in the same camera frame. In white: the naive forward kinematics, for comparison. Best viewed in color.

*This research was supported in part by National Science Foundation grants IIS-1205249, IIS-1017134, EECS-0926052, the Office of Naval Research, the Okawa Foundation, and the Max-Planck-Society.

[1] Autonomous Motion Department at the Max Planck Institute for Intelligent Systems, Tübingen, Germany. Email: first.lastname@tuebingen.mpg.de

[2] Computational Learning and Motor Control lab at the University of Southern California, Los Angeles, CA, USA.

main sources of error along the kinematic chain by explicitly modeling and estimating the biases of the joint measurements, and a time-varying 6-DoF transform describing a correction of the extrinsic calibration of the camera with respect to the robot. The algorithm we derive is computationally efficient and well-suited for real-time implementation. Our code is publicly available (https://github.com/bayesian-object-tracking) and can be used off-the-shelf on any robot given its kinematic model.

For experimental validation, we collect a dataset from two humanoid robotic platforms. It covers a range of challenging conditions, including fast arm and head motion as well as large, long-term occlusions. We also modify the original data to simulate large joint biases. We demonstrate the robustness and accuracy of our system quantitatively and qualitatively compared to multiple baselines. We make this dataset and evaluation code public to allow evaluation and comparison of this and other methods.

To summarize, our contributions are: (i) the description of a probabilistic model and a computationally-efficient algorithm (Sections III and IV); (ii) a practically useful, real-time implementation that we make publicly available; (iii) experimental validation of its robustness and accuracy (Sections V and VI); and (iv) the release of a new dataset for quantitative evaluation.

II. RELATED WORK

Inaccurate and uncertain hand-eye coordination is very common for robotics systems. This problem has therefore been considered frequently in the robotics community.

One approach is to calibrate the transformation between hand and eye offline, prior to performing the manipulation task. Commonly, the robot is required to hold and move specific calibration objects in front of its eyes [1], [2], [3]. This can however be tedious, time-consuming and the calibrated parameters may degrade over time so that the process has to be repeated. Instead, our approach is to continuously track the arm during a manipulation task, which is robust against drifting calibration parameters or against online effects, such as cable stretch due to increased load or contact with the environment.

One possible solution is the use of fiducial markers on specific parts of the robot [4]. Markers have the advantage of being easy to detect but the disadvantage of limiting the arm configurations to keep the markers always in view, and requiring to precisely know their position relative to the robot's kinematic chain. As an alternative to markers, different 2D visual cues can be used to track the entire arm, at the cost of higher computational demand, e.g. texture, gradients or silhouettes, which are also often used in general object tracking [5], [6], [7], [8]. Instead, we choose as visual sensor a depth camera, which readily provides geometric information while being less dependent on illumination effects.

In this paper, we are particularly interested in the question of how to leverage multi-modal sensory data from e.g. proprioception as available in a robotic system. In the following, we review work from the class of marker-less, model-based, multi-modal tracking methods that estimate the configuration of articulated objects online and in real-time, assuming access to joint encoder readings.

Many formulate this problem as the minimization of an objective function for each new incoming frame, and typically use the solution from the previous time step as initialization.

Klingensmith *et al.* [9] present a simple but computationally efficient articulated *Iterative Closest Point* (ICP) [10] method to estimate the joint encoder bias of their robot arm. The objective function is defined in terms of distance between randomly sampled 3D points on the robot model and the point cloud from a depth camera. This is minimized by exploiting the pseudo-inverse of the kinematic Jacobian for computing the gradient.

Pauwels *et al.* [11] consider the problem of simultaneously estimating the pose of multiple objects and the robot arm while it is interacting with them; all relative to a freely moving RGB-D camera. The objective is defined according to the agreement between a number of visual cues (depth, motion) when comparing observations with rendered images given the state. Instead of estimating the robot joint configuration, the authors consider the arm as a rigid object given the encoder values, which are assumed to be precise. To cope with remaining error in the model, robot base and end-effector are considered single objects with the kinematics being enforced through soft-constraints.

There is a related family of methods which optimize for point estimates as well, but additionally consider a model of the temporal evolution the state, and combine them within filtering-based algorithms.

Krainin *et al.* [12] propose a method for in-hand modeling of objects, which requires an estimate of the robot arm pose to be able to segment the hand from the object. They perform articulated ICP similar to [9], and use the result as a measurement of joint angles in a Kalman filter.

Hebert *et al.* [13] estimate the pose of the object relative to the end-effector, and the offset between the end-effector pose according to forward kinematics and visual data. They consider a multitude of cues from stereo and RGB data, such as markers on the hand, the silhouette of object and arm, and 3D point clouds. They employ articulated ICP similar to [9] or 2D signed distance transforms similar to [7] for optimization. An *Unscented Kalman Filter* fuses the results.

Schmidt *et al.* [14] propose a robot arm tracking method based on an *Extended Kalman Filter*. They estimate the bias of the joint encoders similar to [9]. Additional to agreement with depth images, they include physics-based constraints into their optimization function, to penalize interpenetration between object and robot as well as disagreement between contact sensors and the estimate.

In contrast to the filtering-based methods described above, our approach is to model the acquisition of the actual, physical measurements, rather than the uncertainty associated to an optimization result. Probability theory then provides a well-understood framework for fusing the different information sources. This leads to a method with few parameters which are intuitive to choose, because they are closely related to the physics of the sensors.

The method we propose extends our previous work on visual tracking based on a model of raw depth images [15], [16]. These measurements are highly nonlinear and non-Gaussian, for which particle filters are well suited.

In [15], we propose a method to track the 6-DoF pose of a rigid object given its shape. Our image model explicitly takes into account occlusions due to any other objects, which are common in real-world manipulation scenarios. The structure of our formulation makes it suitable to use a Rao-Blackwellized particle filter [17], which in combination with the factorization of the image model over pixels yields a computationally efficient algorithm. This model can be extended to articulated objects given knowledge of the object's kinematics and shape. The additional difficulty is that joint configurations can be quite high-dimensional (> 30). In [16], we propose a method which alleviates this issue by sampling dimension-wise, and show an application to robot arm tracking.

In this paper we take the latter method further by fusing the visual data with the measurements from the joint encoders, therefore exploiting their complementary nature. Additionally to estimating joint angles and/or biases like [16], [9], [14], [13], [12], we simultaneously estimate the true camera pose relative to the kinematic chain of the robot. Further, differently from most approaches mentioned, we process all joint measurements as they arrive rather than limiting ourselves to the image rate, and we handle the delay in the images. A crucial difference to related work is that we model not only self-occlusion of the arm, but also occlusions due to external objects. Although we take as input fewer cues than other methods [11], [14], [13], we already achieve a remarkable robustness and accuracy in robot arm tracking. The dataset we propose in this paper will enable quantitative comparison to alternative approaches.

III. MODELING

Our goal in robot tracking is to recursively estimate the current joint angles a_t of the robot given the history of depth images z_t and joint angle measurements q_t. The kinematics of the robot and shape of its limbs are assumed to be known, so the joint angles are enough to describe the full configuration of the robot, and in particular the pose of the end effector. We further define a number of auxiliary latent variables, which allow to explain the mismatch between the readings from the joint encoders and the robot configuration observed in the depth images:

– We augment the set of joints in the kinematic model with six extra virtual joints. These do not correspond to physical links, but represent a translation and a rotation between the nominal camera pose (i.e. as specified in the kinematic model) and the true camera pose. The nominal camera pose can be measured in advance, at least roughly, if not provided by the manufacturer.

– At each joint j, there is a bias b_t^j that perturbs the joint measurement q_t^j.

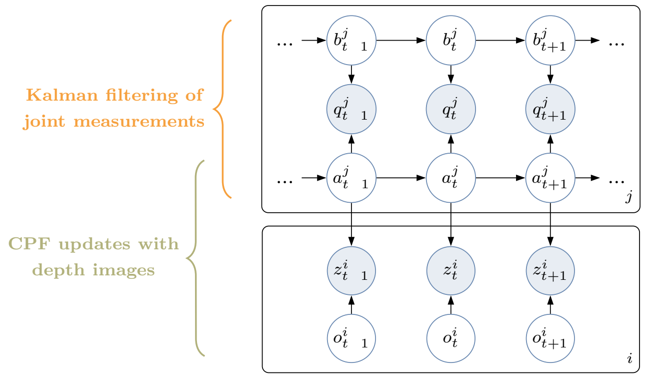


Fig. 2. Bayes network for our model (Section III). Shaded nodes are observed (joint measurements q and depth images z). White nodes are latent (angles a, biases b and occlusions o). t, j and i are indices for discrete time, joints and pixels. Our inference algorithm (Section IV) combines Kalman filtering of joint measurements and image updates based on the CPF [16].

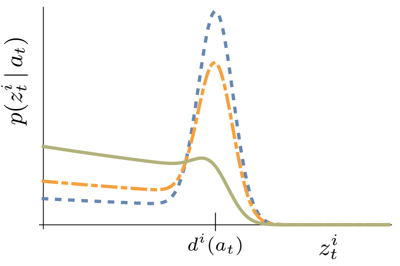


Fig. 3. Distribution of measured depth at one pixel for different occlusion probabilities $p(o^i_t = 1)$, from lower (dotted blue) to higher (solid green).

– For each pixel i of the depth image z_t, a binary variable o_t^i indicates whether an external occluder is present.

The dependences among variables are shown in the Bayes network in Fig. 2. In the remainder of this section, we define the process and measurement distributions that connect them, so as to fully specify our model. In Section IV we provide an algorithm for inference.

A. Camera model

In this paper, we use the same model for depth images as in [15], which factorizes over pixels

$$p(z_t \mid a_t, o_t) = \prod_i p(z_t^i \mid a_t, o_t^i). \tag{1}$$

We consider three sources of error between the rendered and measured depths: (i) inaccuracies in the mesh model, which we model as Gaussian noise; (ii) external occlusions, which cause the measured depth to be lower than the distance to the target object; and (iii) noise in the depth sensor, which we model as a mixture of a Gaussian distribution around the distance to the closest object and a uniform distribution in the range of the sensor. See [15] for the detailed expressions.

Fig. 3 shows an example of the resulting distribution of measured depth at a pixel after marginalizing out the occlusion variable

$$p(z_t^i \mid a_t) = \sum_{o_t^i=\{0,1\}} p(z_t^i \mid a_t, o_t^i) p(o_t^i), \tag{2}$$

for different occlusion probabilities $p(o_t^i = 1)$. It has a peak around the rendered depth $d^i(a_t)$, and a thick tail at lower depths, which accounts for the possibility of an occlusion. The peak becomes less pronounced at higher occlusion probability.

A difference between the present model (Fig. 2) and the model used in [15] is that here the occlusion probabilities are estimated at each step, but not propagated over time. This is a simplification we make to ensure tractability of the more involved filtering problem in the present paper. As will be shown in the experimental section, the resulting algorithm is nevertheless very robust to occlusion.

B. Joint encoder model

We model the joint measurement as the sum of angle, bias, and independent Gaussian noise for each joint j:

$$p(q_t^j \mid a_t^j, b_t^j) = \mathcal{N}(q_t^j \mid a_t^j + b_t^j, \sigma_q^2). \tag{3}$$

C. Angle process model

The angle of each joint j follows a random walk

$$p(a_{t+1}^j \mid a_t^j) = \mathcal{N}(a_{t+1}^j \mid a_t^j, \Delta\sigma_a^2) \tag{4}$$

where Δ is the length of the time step.[1] Parameter σ_a needs to be big enough to capture fast angle dynamics.

This simple model works well in our experiments, likely because of the frequent measurements (every 1-3 ms) with little noise. An interesting question for future work is whether the performance could be improved by using a more complex model, taking into account rigid body dynamics and the control commands sent to the robot.

D. Bias process model

We model the bias such that its variance does not grow indefinitely large in the absence of measurements. A simple linear model that achieves this is the following random walk

$$p(b_{t+1}^j \mid b_t^j) = \mathcal{N}(b_{t+1}^j \mid c^{\Delta} b_t^j, \Delta\sigma_b^2), \quad c < 1, \tag{5}$$

where σ_b denotes the noise standard deviation, c is a parameter specifying how fast the process tends to 0, and Δ is the time step length.[1]

To check that this process would behave as desired, and to gain some intuition on how to choose the parameters, it is helpful to look at its asymptotic behavior in the absence of measurements. We can obtain the asymptotic distribution by taking the distributions at two consecutive time steps

$$p(b_t^j) = \mathcal{N}(b_t^j \mid \mu_*, \sigma_*^2) \tag{6}$$

$$p(b_{t+1}^j) = \mathcal{N}(b_{t+1}^j \mid c^{\Delta}\mu_*, \Delta\sigma_b^2 + c^{2\Delta}\sigma_*^2) \tag{7}$$

and equating their means and variances, which yields

$$\mu_* = 0; \quad \sigma_*^2 = \frac{\Delta}{1 - c^{2\Delta}}\sigma_b^2 \tag{8}$$

for any $\sigma_b > 0$ and $c < 1$. We can see that in the absence of measurements the mean of the bias converges to zero (which is reasonable when there is no data suggesting otherwise), and the variance converges to some constant which depends on the choice of c and σ_b. Equation (8) can help us find meaningful values for these parameters.

[1]The dependence on Δ arises from the integration over time of a continuous process with white noise.

IV. ALGORITHM

Having defined our process and measurement models, our goal now is to find an algorithm for recursive inference. That is, we want to obtain the current belief $p(a_t, b_t \mid z_{1:t}, q_{1:t})$ from the latest measurements z_t and q_t, and the previous belief $p(a_{t-1}, b_{t-1} \mid z_{1:t-1}, q_{1:t-1})$.

Joint and depth measurements are typically generated at different rates in real systems, which makes it useful to be able to separately incorporate a joint measurement or a depth measurement to our belief at any point in time. We assume that we receive joint measurements at a high rate, so we choose the time interval between joint measurements to be the basic time step at which we want to produce estimates. Further, we have to cope with a delay in the depth image, as explained in Section IV-C.

A. Filtering joint measurements

Defining $h_t = \{z_{1:t-1}, q_{1:t-1}\}$ for brevity, the incorporation of a joint measurement can be written as

$$p(a_t, b_t \mid q_t, h_t) \propto p(q_t \mid a_t, b_t) \cdot \int_{a_{t-1}, b_{t-1}} p(a_t \mid a_{t-1}) p(b_t \mid b_{t-1}) p(a_{t-1}, b_{t-1} \mid h_t) \tag{9}$$

where we used our assumption that the angle process (4) and the bias process (5) are independent. The models involved in (9) are (3, 4, 5). All of them are linear Gaussian, hence (9) has a closed-form solution, which corresponds to one time step of a *Kalman Filter* (KF) [18]. Furthermore, all these models factorize in the joints. Hence, if the initial belief $p(a_{t-1}, b_{t-1} \mid h_t)$ factorizes too, we can filter with an independent KF for each joint, which greatly improves efficiency. We will see in the following how we keep the belief Gaussian and factorize in the joints at all times, so that this is indeed the case.

B. Updating with depth images

Let $\hat{h}_t = \{q_t, h_t\}$ be the history of measurements before the image update. Each time a depth image is received, we update the belief obtained in (9), i.e. $p(a_t, b_t \mid \hat{h}_t)$, to get the desired posterior $p(a_t, b_t \mid z_t, \hat{h}_t)$.

The goal of this section is to write this update in such a form that we can apply our previous work [16], so as to handle the high-dimensional state efficiently.

We begin by noting the following equalities:

$$p(a_t, b_t \mid z_t, \hat{h}_t) = \frac{p(z_t \mid a_t, b_t, \hat{h}_t) p(a_t, b_t \mid \hat{h}_t)}{p(z_t \mid \hat{h}_t)} \tag{10}$$

$$= \frac{p(z_t \mid a_t, \hat{h}_t) p(a_t \mid \hat{h}_t) p(b_t \mid a_t, \hat{h}_t)}{p(z_t \mid \hat{h}_t)} \tag{11}$$

$$= p(a_t \mid z_t, \hat{h}_t) p(b_t \mid a_t, \hat{h}_t), \tag{12}$$

where we used that our image model does not depend on the bias. According to (12), the desired posterior can be decomposed into two terms. The first is the posterior in the angle only, which we will derive in the following. The second is easily obtained from the prior (9) by conditioning on a_t, which for a Gaussian can be done in closed form [19, p. 90].

1) Posterior in the angle: We can write the posterior in the angle as

$$p(a_t \,|\, z_t, \hat{h}_t) \propto p(z_t \,|\, a_t) p(a_t \,|\, \hat{h}_t), \tag{13}$$

where the first term is the image observation model (1), and the second is readily obtained from the Gaussian prior (9) by marginalizing out b_t [19, p. 90].

Since this update only involves the image z_t and the joint angles a_t, we can solve it in the same manner as we did in [16]. This involves sampling from $p(a_t \,|\, \hat{h}_t)$ dimension-wise, and weighting with the likelihood $p(z_t \,|\, a_t)$. For more details, we refer the reader to [16], [15] – what is important here is that this step creates a set of particles $\{^{(l)}a_t\}_l$ distributed according to (13).

2) Gaussian approximation: After incorporating a depth image, we approximate the particle belief (13) with a Gaussian distribution factorizing in the joints

$$p(a_t \,|\, z_t, \hat{h}_t) \approx \prod_j \mathcal{N}(a_t^j \,|\, \mu_t^j, \Sigma_t^j) \tag{14}$$

Moment matching is well known to be the minimum KL-divergence solution for Gaussian approximations [19, pp. 505-506]. Therefore, we make

$$\mu_t^j = \frac{1}{L}\sum_{l=1}^{L} {}^{(l)}a_t^j \;, \;\; \Sigma_t^j = \frac{1}{L}\sum_{l=1}^{L} ({}^{(l)}a_t^j - \mu_t^j)^2. \tag{15}$$

3) Full posterior: Since we approximated both terms in (12) by Gaussians which factorize in the joints, the full posterior is of the same form

$$p(a_t, b_t \,|\, z_t, \hat{h}_t) = \prod_j \mathcal{N}(a_t^j, b_t^j \,|\, m_t^j, M_t^j). \tag{16}$$

Hence, we can continue to filter joint measurements using independent KFs for each joint, as mentioned in Section IV-A. The parameters m_t^j and M_t^j are easily obtained using standard Gaussian manipulations, e.g. 'completing the square' in the exponent [19, p. 86].

Approximating distributions by factorized distributions is a common practice in machine learning to ensure tractability, in particular in variational inference. In our case, this factorization allows us to have n KFs on a 2-dimensional state each (with complexity $O(n)$), instead of one KF on a $2n$-dimensional state (with complexity $O(n^3)$).

C. Taking into account image delay

An additional difficulty is that images are very data heavy, which often leads to delays in acquisition and transmission. This means that at time t we might receive an image with time stamp $t' < t$. However, all the joint angles between t' and t have been processed already, and our current belief is $p(a_t, b_t \,|\, q_t, h_t)$. Our solution to this problem is to maintain a buffer of beliefs and joint measurements. When an image is received, we find the belief $p(a_{t'}, b_{t'} \,|\, q_{t'}, h_{t'})$ with the appropriate time stamp, and incorporate the image $z_{t'}$ into this belief according to Section IV-B. Once this is done, we re-filter the joint measurements in the buffer which have a time stamp $> t'$ according to Section IV-A to obtain the current belief. Therefore, the filtering of joint angles has to be extremely fast, which is possible due to the factorization in the joints.

D. Efficiency and implementation

We implemented our tracker as two filters executed in parallel at different rates, which can be reinitialized from each other's belief at any time as described above. The components which make the proposed method real-time capable are:

– the factorization in the pixels of our depth measurement model [15];

– the use of the *Coordinate Particle Filter* [16], which allows to filter in the high dimensional joint space of the robot with relatively few particles;

– the factorization of the belief in the joints as described in Section IV-B, which allows to apply independent Kalman filters for each joint; and

– our GPU implementation for computing the depth image likelihoods for all particles in parallel [20].

Our code can be used off-the-shelf by providing the robot's model in Unified Robot Description Format. Other nice features are the optional automatic injection of the virtual joints into the robot model, and that it is easily configurable to account for a constant offset in the image time stamp.

V. EXPERIMENTAL SETUP

To evaluate our approach towards robust and accurate robot tracking, we recorded data from two different robotic platforms. This section describes these platforms, the type of data, baselines and evaluation measures. We make this dataset public, together with the robot models and evaluation code, so as to allow comparison among methods and reproduction of the results here.

A. Robotic platforms

We recorded data on two different robotic platforms. They are both fixed-base dual-arm manipulation platforms equipped with two three-fingered Barrett Hands and an RGB-D camera (Asus Xtion) mounted on an active head. They differ in the source of error that leads to an inaccurate hand-eye coordination.

Apollo (Fig. 1a) is equipped with two 7-DoF Kuka LWR IV arms with very accurate joint encoders. The active humanoid head on which the RGB-D camera is mounted has a mechanism consisting of two four-bar linkages that are connected at the head and generate a coupled motion. We use an approximation to easily compute the 3 rotary DoF of the neck from linear joint encoders. This causes non-linear, configuration-dependent error in the pose of the head-mounted camera.

The ARM robot (Fig. 1b) consists of two 7-DoF Barrett WAM arms that are actuated using a cable-driven mechanisms with motors and encoders in the shoulder. Upon contact with the environment or during lifting a heavy object, the resulting cable stretch is not observable through the encoders and therefore leads to a time-varying bias on the joint angles. The active head consists of two stacked pan-tilt units with simple kinematics and accurate encoders.

While the dominant error for Apollo comes from the inaccurate kinematic model from the base to the camera, hand-eye coordination on the ARM robot is mostly perturbed by biases in the joint encoder readings. With this kind of data, we cover the most important errors that cause inaccurate hand-eye coordination in many robotics systems.

B. *Data from Apollo*

The data recorded on Apollo consists of time-stamped RGB-D images (although our method uses only depth), and measurements from the joint encoders. We also measured the pose of the head and end effector using a VICON motion-capture system. After a calibration procedure to align the camera frame to the VICON system, this provides ground truth poses of the end effector with regards to the camera.

The dataset has seven 60-second-long sequences recorded while Apollo was in gravity compensation, and where its right arm and/or head is moved by a person. The sequences include fast motion (labeled with the '++' suffix) and simultaneous arm and head motion ('both'). One of the sequences contains heavy, long-term occlusions of the moving arm ('occlusion'). Another contains a long period in which the arm is out of the camera's field of view ('in/out/in'). In five additional 30-second-long sequences, Apollo performs sinusoidal motion (prefix 's-') of lower, upper and full arm, head motion, and simultaneous head and arm motion.

As mentioned in Section V-A, the kinematics of the head are quite inaccurate, while they are precise in the arm. Therefore, we extend this dataset by simulating bias in the joint sensors of the arm. We do this by adding an offset to the measured joint angles, while leaving the RGB-D images and ground-truth poses intact. One version of the simulated dataset has a constant offset of 8.6 degrees in each arm joint. Another has a time-varying offset on each joint, consisting of smooth steps of alternating sign and 5 degrees of amplitude.

C. *Data from ARM*

In the ARM robot, the joint measurements are contaminated with real noise and bias, but we do not have ground truth labels. We have several sequences of depth images and joint measurements of the robot moving its two arms in sinusoidal waves at different frequencies. We use this data for qualitative evaluation.

D. *Performance measures*

Our method produces estimates of joint angles, and a correction of the camera pose with regards to the robot. In particular, this provides the pose of the estimated end effector (the hand) in the estimated camera frame. We compare this pose to the ground-truth hand-to-camera pose provided by the VICON system. We use as performance measures the translational and angular error.

These methods will be compared in the following section:

– *Encoders only*: a baseline that predicts the hand-to-camera pose by simply applying forward kinematics on the kinematic model, using the raw measured angles.

– *Camera offset only*: a variant of our method that assumes the joint measurements contain no bias, and only estimates the virtual joints describing the camera offset, i.e. performs online extrinsic calibration of the camera with regards to the kinematic chain. It uses both joint measurements and depth images.

– *Full fusion*: our full method, with parameters allowing to express large biases. It fuses joint measurements and vision at every joint, and also estimates virtual joints.

– *Vision only*: a variant of our method that relies only on images, similar to the experiments in [16].

To summarize the performance of each method in the dataset, we use box plots with one bar per method and sequence. Each bar in the plot represents statistics of the translational or angular error obtained by the method on the sequence, taken at regular time intervals, aggregated over the length of the sequence, and further aggregated over 10 runs. See for example Fig. 4. The black ticks and limits of the colored bars indicate the 1st, 25th, 50th, 75th and 99th percentiles.

E. *Parameters*

The parameters of the image observation model are as in [15]; they are not specific to robot tracking. The new parameters in the proposed method are the noise σ_q in the encoder model (3), the noise σ_a in the angle process model (4), and finally the parameters c and σ_b of the bias process (5). All these parameters have a physical meaning and were chosen in an intuitive manner with only minimal tuning.

There is a trade-off between the magnitude of bias the method can absorb, and the accuracy/smoothness of the estimate. Nevertheless, we used the same parameter set for all the quantitative experiments in the following section. In practice, one could achieve some improvement in accuracy by adapting the parameters to the situation, e.g. reducing σ_b or c if we know the bias of the encoders to be low.

VI. EVALUATION

A. *Quantitative evaluation on real data*

In this experiment, we show the benefit of estimating the virtual joints to correct inaccuracies in the head kinematics. We use real annotated data from Apollo.

Because of the properties of this robot described above, it is safe to assume accurate joint measurements, i.e. no bias, and only estimate the virtual joints. This corresponds to the camera-offset-only method. We compare it to the encoders-only baseline in Fig. 4. Clearly, estimating the camera offset decreases the error significantly. The 75th-percentile error without estimating the camera offset is in the order of a few centimeters. The camera offset correction allows to reduce this error to a few millimeters. The 99th percentile is still relatively high after correction. This is because it takes some time in the beginning of a sequence for the camera offset to converge to the correct value. But once this is achieved, we consistently obtain a low error.

The full-fusion method estimates the bias in the joint angles in addition to the camera offset. We see in Fig. 4 that its accuracy is very similar to the camera-offset-only method. This is expected, due to the absence of biases in the joints of the Apollo robot.

B. *Quantitative evaluation on simulated biases*

This section shows how our system is capable of dealing with strong, time-varying biases on the joint measurements in a range of conditions. We use the same data as for the previous experiment, except that here we corrupted the joint measurements, as described in Section V-B.

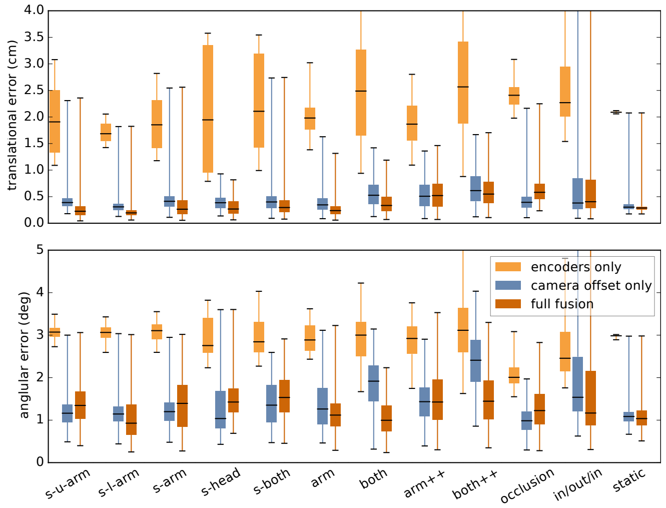


Fig. 4. Performance on **real** data with accurate joint encoders.

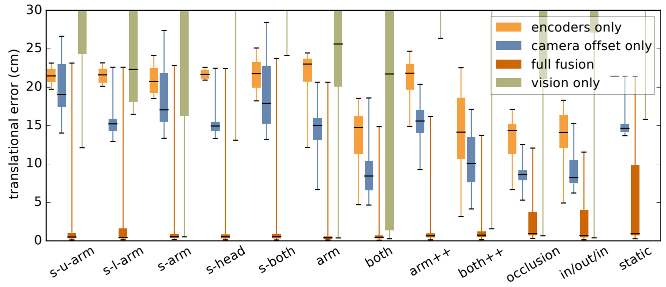


Fig. 5. Performance on data with **simulated constant** biases.

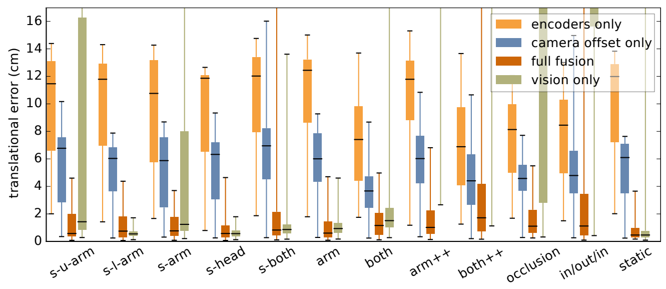


Fig. 6. Performance on data with **simulated time-varying** biases.

As we can see in Fig. 5, the error in the end-effector position can be as high as 25 cm when applying forward kinematics to the biased measurements. While estimating the camera offset decreases this error by several centimeters, we need the full fusion to bring it down to the order of the millimeters. As shown in Fig. 7a, the full-fusion method requires some time to correct the large bias, but once it converged, it consistently yields precise estimates.

The case of the time-varying bias (Fig. 6) is more challenging, because of the sudden bias changes of up to 10 degrees. In the time-series plots in Fig. 7 we can see how the fusion filter needs some time to adapt its estimate after each change, so its error is higher during this transition.

It is interesting to compare the behavior of the fusion tracker to the purely visual one. The latter does not use joint measurements, so its performance does not suffer from perturbations in them, see e.g. Fig. 7d. However, it easily loses track during occlusion, fast motions, or when the robot arm is out of view. In contrast, the estimate of the fusion tracker stays at least as accurate as the encoders-only method, even when the visual information is not reliable, e.g. during the strong occlusions in Fig. 7e and 7c, see Fig. 1c for an example frame. Similarly, when the arm goes out of view (Fig. 7b) the fusion tracker's estimates are pulled towards the joint measurement, so the error increases. But then it can recover when the arm is within view again.

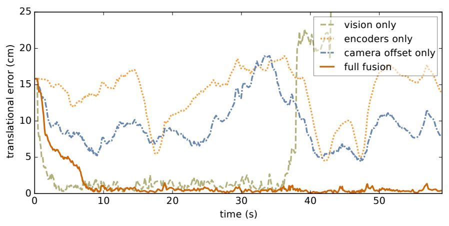
C. *Qualitative evaluation*

Fig. 1c shows one shot of the Apollo sequence with strong occlusions and large simulated constant bias. We can see that our estimate remains accurate despite most of the arm being covered by a person. Fig. 1d shows real depth data from the ARM robot. It gives an idea of the amount of error produced by real bias. More examples of qualitative evaluation can be found in the supplementary video (https://youtu.be/YNP9UCx6Wa4) showing robust and accurate tracking during simultaneous arm and head motion, fast arm motion and when the arm is going in and out of view.

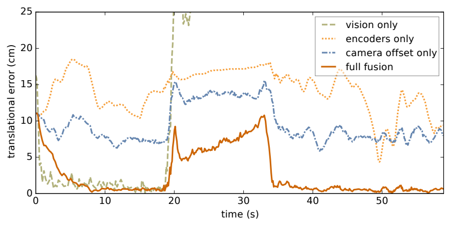
VII. CONCLUSION

We proposed a probabilistic filtering algorithm which fuses joint measurements with depth images to achieve robust and accurate robot arm tracking. The strength of our approach lies in modeling the measurements in an intuitive, generative way that is realistic enough to achieve high precision, while keeping in mind tractability. In our case, this implied introducing and explicitly estimating auxiliary latent variables such as pixel occlusions, encoder biases and camera offset, which enable a good fit of the data. Some principled approximations, as well as building on previous work such as [16], [20], made it possible to derive a computationally efficient algorithm and real-time implementation.

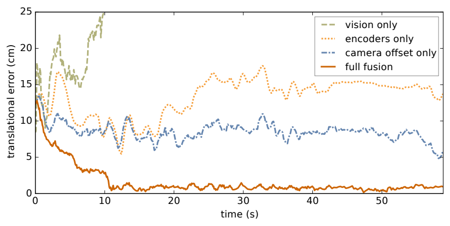
In this paper, we showed that our method performs quantitatively well under the challenging conditions of the dataset we propose, including fast motion, significant and long-term occlusions, time-varying biases, and the robot arm getting in



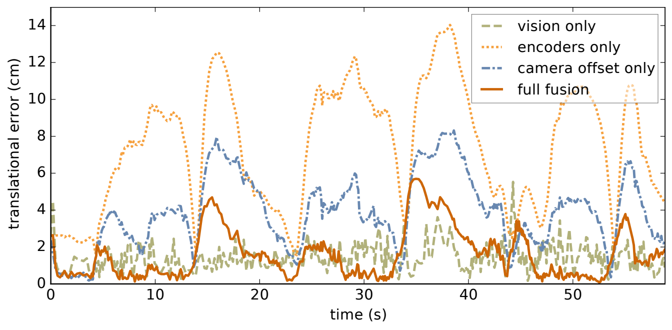
(a) simultaneous head and arm motion



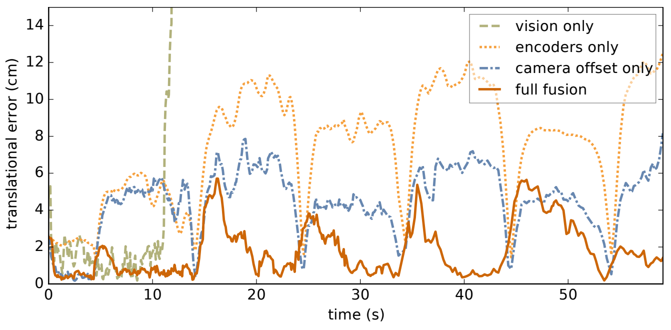
(b) arm goes out of view and back



(c) strong occlusions



(d) simultaneous head and arm motion



(e) strong occlusions

Fig. 7. Example run on sequences with **constant** (top row) and **time-varying** (bottom row) simulated biases.

and out of view. Further, we have already demonstrated how this method can be integrated into an entire manipulation system that simultaneously tracks robot arm and object to enable pick and place tasks in uncertain and dynamic environments [21].

Our system is already very robust when tracking the arm using only depth images and joint measurements. An interesting direction for future work is to fuse data from haptic sensors, which may further improve performance when simultaneously estimating object and arm pose during manipulation.

References

[1] P. Pastor, M. Kalakrishnan, J. Binney, J. Kelly, L. Righetti, G. Sukhatme, and S. Schaal, "Learning task error models for manipulation," in *IEEE Intl Conf on Robotics and Automation*, May 2013, pp. 2612–2618.

[2] K. Pauwels and D. Kragic, "Integrated on-line robot-camera calibration and object pose estimation," in *IEEE Intl Conf on Robotics and Automation*, May 2016, pp. 2332–2339.

[3] V. Pradeep, K. Konolige, and E. Berger, "Calibrating a multi-arm multi-sensor robot: A bundle adjustment approach," in *Intl Symposium on Experimental Robotics (ISER)*, New Delhi, India, Dec 2010.

[4] N. Vahrenkamp, C. Böge, K. Welke, T. Asfour, J. Walter, and R. Dillmann, "Visual servoing for dual arm motions on a humanoid robot," in *IEEE-RAS Intl Conf on Humanoid Robots*, 2009, pp. 208–214.

[5] C. Choi and H. I. Christensen, "Real-time 3D model-based tracking using edge and keypoint features for robotic manipulation," in *IEEE Intl Conf on Robotics and Automation*, May 2010, pp. 4048–4055.

[6] D. Kragic, A. T. Miller, and P. K. Allen, "Real-time tracking meets online grasp planning," in *IEEE Intl Conf on Robotics and Automation*, vol. 3, 2001, pp. 2460–2465.

[7] X. Gratal, J. Romero, J. Bohg, and D. Kragic, "Visual servoing on unknown objects," *Mechatronics*, vol. 22, no. 4, pp. 423 – 435, 2012.

[8] S. Hinterstoisser, C. Cagniart, S. Ilic, P. F. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of textureless objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 876–888, 2012.

[9] M. Klingensmith, T. Galluzzo, C. Dellin, M. Kazemi, J. A. D. Bagnell, and N. Pollard , "Closed-loop servoing using real-time markerless arm tracking," in *IEEE Intl Conf on Robotics and Automation (Humanoids Workshop)*, May 2013.

[10] P. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[11] K. Pauwels, V. Ivan, E. Ros, and S. Vijayakumar, "Real-time object pose recognition and tracking with an imprecisely calibrated moving RGB-D camera," in *IEEE/RSJ Intl Conf on Intelligent Robots and Systems*, Chicago, Illinois, 2014.

[12] M. Krainin, P. Henry, X. Ren, and D. Fox, "Manipulator and object tracking for in-hand 3D object modeling," *The Intl Journal of Robotics Research*, 2011.

[13] P. Hebert, N. Hudson, J. Ma, T. Howard, T. Fuchs, M. Bajracharya, and J. Burdick, "Combined shape, appearance and silhouette for simultaneous manipulator and object tracking," in *IEEE Intl Conf on Robotics and Automation*, 2012.

[14] T. Schmidt, K. Hertkorn, R. A. Newcombe, Z. Marton, M. Suppa, and D. Fox, "Depth-based tracking with physical constraints for robot manipulation," in *IEEE Intl Conf on Robotics and Automation*, May 2015, pp. 119–126.

[15] M. Wüthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic object tracking using a range camera," in *IEEE/RSJ Intl Conf on Intelligent Robots and Systems*, 2013.

[16] M. Wüthrich, J. Bohg, D. Kappler, P. C., and S. S., "The Coordinate Particle Filter - a novel Particle Filter for high dimensional systems," in *IEEE Intl Conf on Robotics and Automation*, May 2015.

[17] A. Doucet, N. de Freitas, K. Murphy, and S. Russell, "Rao-Blackwellised particle filtering for dynamic Bayesian networks," in *Proc of the 16th Conf on Uncertainty in Artificial Intelligence*, 2000, pp. 176–183.

[18] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME - Journal of Basic Engineering*, no. 82 (Series D), pp. 35–45, 1960.

[19] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[20] C. Pfreundt, "Probabilistic object tracking on the GPU," Master's thesis, Karlsruhe Institute of Technology, Mar. 2014.

[21] J. Bohg, D. Kappler, F. Meier, N. Ratliff, J. Mainprice, J. Issac, M. Wüthrich, C. Garcia Cifuentes, V. Berenz, and S. Schaal, "Interlocking perception-action loops at multiple time scales - a system proposal for manipulation in uncertain and dynamic environments," in *Intl Workshop on Robotics in the 21st Century: Challenges and Promises*, Sep 2016.