Single-view robot pose and joint angle estimation via render & compare

Yann Labbé [1] Justin Carpentier [1] Mathieu Aubry [2] Josef Sivic [1,3]
[1] ENS/Inria [2] LIGM, ENPC [3] CIIRC CTU
https://www.di.ens.fr/willow/research/robopose

Abstract

We introduce RoboPose, a method to estimate the joint angles and the 6D camera-to-robot pose of a known articulated robot from a single RGB image. This is an important problem to grant mobile and itinerant autonomous systems the ability to interact with other robots using only visual information in non-instrumented environments, especially in the context of collaborative robotics. It is also challenging because robots have many degrees of freedom and an infinite space of possible configurations that often result in self-occlusions and depth ambiguities when imaged by a single camera. The contributions of this work are three-fold. First, we introduce a new render & compare *approach for estimating the 6D pose and joint angles of an articulated robot that can be trained from synthetic data, generalizes to new unseen robot configurations at test time, and can be applied to a variety of robots. Second, we experimentally demonstrate the importance of the robot parametrization for the iterative pose updates and design a parametrization strategy that is independent of the robot structure. Finally, we show experimental results on existing benchmark datasets for four different robots and demonstrate that our method significantly outperforms the state of the art. Code and pre-trained models are available on the project webpage [1].*

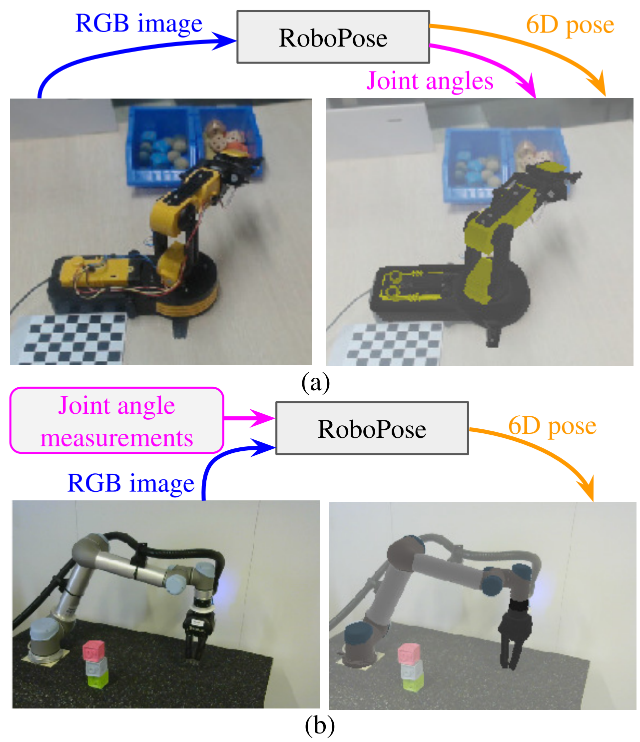


Figure 1: **RoboPose.** (a) Given a single RGB image of a known articulated robot in an unknown configuration (left), RoboPose estimates the joint angles and the 6D camera-to-robot pose (rigid translation and rotation) providing the complete state of the robot within the 3D scene, here illustrated by overlaying the articulated CAD model of the robot over the input image (right). (b) When the joint angles are known at test-time (e.g. from internal measurements of the robot), RoboPose can use them as an additional input to estimate the 6D camera-to-robot pose to enable, for example, visually guided manipulation without fiducial markers.

1. Introduction

The goal of this work is to recover the state of a known articulated robot within a 3D scene using a single RGB image. The robot state is defined by (i) its 6D pose, i.e. a 3D translation and a 3D rotation with respect to the camera frame, and (ii) the joint angle values of the robot's articulations. The problem set-up is illustrated in Figure 1. This is an important problem to grant mobile and itinerant autonomous systems the ability to interact with other robots using only visual information in non-instrumented environments. For instance, in the context of collaborative tasks between two or more robots, having knowledge of the pose and the joint angle values of all other robots would allow better distribution of the load between robots involved in the task [5].

The problem is, however, very challenging because robots can have many degrees of freedom (DoF) and an infinite space of admissible configurations that often result in self-occlusions and depth ambiguities when imaged by a single camera. The current best performing methods for this problem [28, 61] use a deep neural network to localize in the image a fixed number of pre-defined keypoints (typically located at the articulations) and then solve a 2D-to-3D optimization problem to recover the robot 6D pose [28] or pose

[1] Inria Paris and Département d'informatique de l'ENS, École normale supérieure, CNRS, PSL Research University, 75005 Paris, France.
[2] LIGM, École des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-vallée, France.
[3] Czech Institute of Informatics, Robotics and Cybernetics at the Czech Technical University in Prague.

and configuration [61]. For rigid objects, however, methods based on 2D keypoints [34, 3, 7, 6, 45, 52, 23, 50, 44, 43, 18] have been recently outperformed by *render & compare* methods that forgo explicit detection of 2D keypoints but instead use the entire shape of the object by comparing the rendered view of the 3D model to the input image and iteratively refining the object's 6D pose [59, 31, 25]. Motivated by this success, we investigate how to extend the *render & compare* paradigm for articulated objects. This presents significant challenges. First, we need to estimate many more degrees of freedom than the sole 6D pose. Articulated robots we consider in this work can have up to 15 degrees of freedom in addition to their 6D rigid pose in the environment. Second, the space of configurations is continuous and hence there are infinitely many configurations in which the object can appear. As a result, it is not possible to see all configurations during training and the method has to generalize to unseen configurations at test time. Third, the choice of transformation parametrization plays an important role for 6D pose estimation of rigid objects [31] and finding a good parametrization of pose updates for articulated objects is a key technical challenge.

Contributions. To address these challenges, we make the following contributions. First, we introduce a new *render & compare* approach for estimating the 6D pose and joint angles of an articulated robot that can be trained from synthetic data, generalizes to new unseen robot configurations at test time, and can be applied to a large variety of robots (robotic arms, bi-manual robots, etc.). Second, we experimentally demonstrate the importance of the robot pose parametrization for the iterative pose updates and design an effective parametrization strategy that is independent of the robot. Third, we apply the proposed method in two settings: (i) with known joint angles (e.g. provided by internal measurements from the robot such as joint encoders), only predicting the camera-to-robot 6D pose, and (ii) with unknown joint angles, predicting both the joint angles *and* the camera-to-robot 6D pose. We show experimental results on existing benchmark datasets for both settings that include a total of four different robots and demonstrate significant improvements compared to the state of the art.

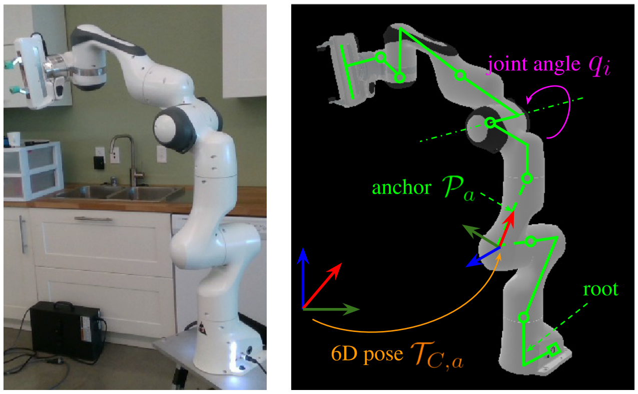
2. Related work

6D pose estimation of rigid objects from RGB images [46, 33, 34] is one of the oldest problems in computer vision. It has been successfully approached by estimating the pose from 2D-3D correspondences obtained via local invariant features [34, 3, 7, 6], or by template-matching [14]. Both these strategies have been revisited using convolutional neural networks (CNNs). A set of sparse [45, 52, 23, 50, 44, 43, 18] or dense [56, 41, 49, 59] features is detected on the object in the image using a CNN and the resulting 2D-to-3D correspondences are used to recover the camera pose using PnP [29]. The best performing methods for 6D pose estimation from RGB images are now based on variants of the *render & compare* strategy [31, 35, 38, 25, 59] and are approaching the accuracy of methods using depth as input [16, 15, 31, 25].

Hand-eye calibration (HEC) [17, 13] methods recover the 6D pose of the camera with respect to a robot. The most common approach is to detect in the image fiducial markers [10, 9, 39] placed on the robot at known positions. The resulting 3D-to-2D correspondences are then used to recover the camera-to-robot pose using *known* joint angles and the kinematic description of the robot by solving an optimization problem [40, 19, 57]. Recent works have explored using CNNs [27, 28] to perform this task by recognizing 2D keypoints at specific robot parts and using the resulting 3D-to-2D correspondences to recover the hand-eye calibration via PnP. The most recent work in this direction [28] demonstrated that such learning-based approach could replace more standard hand-eye calibration methods [54] to perform online calibration and object manipulation [53]. Our *render & compare* method significantly outperforms [28] and we also demonstrate that our method can achieve a competitive accuracy without requiring known joint angles at test time.

Depth-based pose estimation of articulated objects. Previous work on this problem can be split into three classes. The first class of methods aims at discovering properties of the kinematic chain through active manipulation [21, 22, 11, 36] using depth as input and unlike our approach cannot be applied to a single image. The second class of methods aims at recovering all parameters of the kinematic chain from a single RGBD image, including the joint angles, without knowing the specific articulated object [30, 58, 2, 60]. In contrast, we focus on the set-up with a known 3D model, e.g. a specific type of a robot. The third class of methods, which is closest to our set-up, considers pose and joint angle estimation [37, 8, 42] for known articulated objects but relies on depth as input and only considers relatively simple kinematic chains such as laptops or drawers where the joint parameters only affect the pose of one part. Others recover joint angles of a known articulated robot part [4, 55] but do not recover the 6D pose of the camera and also rely on depth. In contrast, our method accurately estimates the pose and joint angles of a robotic arm with many degrees of freedom from a single RGB image.

Robot pose and joint angle estimation from an RGB image. To the best of our knowledge, only [61] has addressed a scenario similar to us where the robot pose and joint angles are estimated together from a single RGB image. A set of predefined 2D keypoints is recognized in the image and the 6D pose and joint angles are then recovered by solving a nonlinear non-convex optimization problem. Results are shown on a 4 DoF robotic arm. In contrast, we describe a



(a) (b)

Figure 2: **Problem definition.** Given an RGB image (a) of a known robot, the goal is to recover (b) the 6D pose $\mathcal{T}_{C,a}$ of an anchor part $\mathcal{P}_a$ with respect to the camera frame and all the joint angles q_i of the known robot kinematic description (in green).

new *render & compare* approach for this problem, demonstrate significantly improvements in 3D accuracy and show results on robots with up to 15 DoF.

3. Approach

We present our *render & compare* framework to recover the state of a robot within a 3D scene given a single RGB image. We assume the camera intrinsic parameters, the CAD model and kinematic description of the robot are known. We start by formalizing the problem in Section 3.1. We then present an overview of our approach in Section 3.2 and explain our training in Section 3.3. Finally, we detail the key choices in the problem parametrization in Section 3.4.

3.1. Problem formalization

Our notations are summarized in Figure 2. We consider a known robot composed of *rigid parts* $\mathcal{P}_0$,...,$\mathcal{P}_N$ whose 3D models are known. An articulation, or *joint*, connects a parent part to a child part. Given the *joint angle* q_i of the i-th joint, we can retrieve the relative 6D transformation between the parent and child reference frames. Note that for simplicity we only consider revolute joints, i.e. joints parametrized by a single scalar angle of rotation q_i, but our method is not specific to this type of joints. The rigid parts and the joints define the *kinematic tree* of the robot. This kinematic description can be used to compute the relative 6D pose between any two parts of the robot. In robotics, the full state of a robot $\mathcal{S}$ is defined by the joint angles and the 6D pose of the *root* of the kinematic tree. Defining the 6D pose of the robot with respect to the root (whose pose is independent of the joint angles since it is not a child of any joint) is a crucial choice in the parametrization of the problem, but also arbitrary, since an equivalent kinematic tree could be defined using any part as the root. We instead define the full state of the robot by (i) the selection of an *anchor part* $\mathcal{P}_a$, (ii) the 6D pose of the anchor with respect to the camera $\mathcal{T}_{C,a}$, and (iii) the joint angles $q = (q_1, ..., q_D) \in \mathbb{R}^D$, where D is the number of joints. Note the anchor part can change across iterations of our approach. We discuss the choice of the anchor in Section 3.4 and experimentally demonstrate it has an important influence on the results.

3.2. Render & compare for robot state estimation

We now present our iterative deep *render & compare* framework, illustrated in Figure 3. We iteratively refine the state estimate as follows. First, given a current estimate of the state $\mathcal{S}^k$ we render an RGB image of the robot $\mathcal{R}(S^k)$ and the mask of the anchor part. We then apply a deep refiner network that takes as input crops of the rendered image and the input RGB image I of the scene. It outputs a new state of the robot $\mathcal{S}^{k+1} = f_\theta(\mathcal{S}^k, I)$ to attempt to match the ground truth state $\mathcal{S}^{gt}$ of the observed robot. Unlike prior works that have used *render & compare* strategies for estimating the 6D pose of rigid objects [31, 59, 25], our method does not require a coarse pose estimate as initialization.

Image rendering and cropping. To render the image of the robot we use a fixed focal length (defining an intrinsic camera matrix) during training. The rendering is fully defined by the state of the robot and the camera matrix. Instead of giving to the refiner network the full image and the rendered view, we focus the inputs on the robot by cropping the images as follows. We project the centroid of the rendered robot in the image, consider the smallest bounding box of aspect ratio 4/3 centered on this point that encloses the projected robot and increase its size by 40% (see details in the appendix [26]). This crop depends on the projection of the robot to the input image that varies during training, and thus provides an augmentation of the effective focal length of the virtual cropped cameras. Hence, our method can be applied to cameras with different intrinsics at test time as we show in our experiments.

Initialization. We initialize the robot to a state $\mathcal{S}^0$ defined by the joint configuration q^0 and the pose $\mathcal{T}^0_{C,a}$ of the anchor part a with respect to the camera C. At training time we define $\mathcal{S}^0$ using perturbations of the ground truth state. At test time we initialize the joints to the middle of the joint limits, and the initial pose $\mathcal{T}^0_{C,a}$ so that the frame of the robot base is aligned with the camera frame and the 2D bounding box defined by the projection of the robot model approximately matches the size of image. More details are given in the appendix [26].

Refiner and state update. At iteration k, the refiner predicts an update Δq^k of the joint angles q^k (composed of one scalar angle per joint), such that

$$q^{k+1} = q^k + \Delta q^k, \tag{1}$$

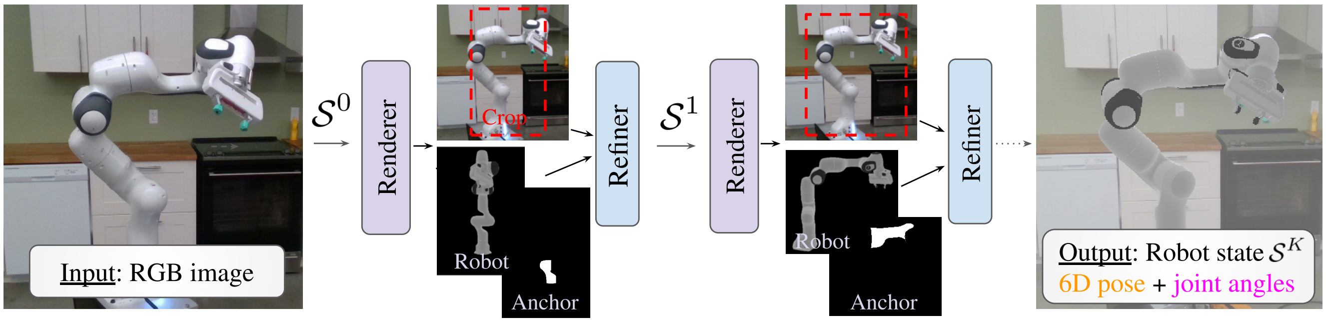


Figure 3: **RoboPose overview.** Given a single input RGB image, the state S (6D camera-to-robot pose and joint angles) of the robot is iteratively updated using renderer and refiner modules to match the input image. The refinement module takes as input the cropped observed image and a rendering of the robot as well as the mask of an anchor part. The anchor part is used for updating the rigid 6D pose of the robot while the rest of the parts are updated by changing their joint angles. Note that the anchor part is changing across iterations making the refinement more robust.

and an update $\Delta\mathcal{T}^k$ of the current 6D pose $\mathcal{T}^k_{C,a}$ of the anchor part, such that

$$\mathcal{T}^{k+1}_{C,a} = \mathcal{T}^k_{C,a} \circ \Delta\mathcal{T}^k, \quad (2)$$

where we follow DeepIM [31]'s parametrization for pose update $\Delta\mathcal{T}^k$. This parametrization disentangles rotation and translation prediction but crucially depends on the choice of a *reference point* we call O. In DeepIM this point is simply taken as the center of the reference frame of the rigid object but there is not such a natural choice of reference point for articulated objects, which have multiple moving parts. We discuss several possible choices of the reference point O in Sec. 3.4 and demonstrate experimentally it has an important impact on the results. In particular, we show that naively selecting the reference frame of the root part is sub-optimal.

3.3. Training

In the following, we describe our loss function, synthetic training data, implementation details and discuss how to best use known joint angles if available.

Loss function. We train our refiner network using the following loss:

$$\mathcal{L}(\theta) = \sum_{k=0}^{K-1} \mathcal{L}_a(\mathcal{T}^k_{C,a}, \Delta\mathcal{T}^k, \mathcal{T}^{gt}_{C,a}) + \lambda\, \mathcal{L}_q(q^k, \Delta q^k, q^{\text{gt}}), \quad (3)$$

where θ are the parameters of the refiner network, K is the maximum number of iterations of the refinement algorithm, $\mathcal{T}^{gt}_{C,a}$ is the ground truth 6D pose of the anchor, q^{gt} are the ground truth joint angles and λ is a hyper-parameter to balance between the 6D pose loss $\mathcal{L}_a$ and the joint angle loss $\mathcal{L}_q$. The 6D pose loss $\mathcal{L}_a$ measures the distance between the predicted 3D point cloud obtained using $\mathcal{T}^k_{C,a}$ transformed with $\Delta\mathcal{T}^k$ and the ground truth 3D point cloud (obtained using $\mathcal{T}^{gt}_{C,a}$) of the anchor $\mathcal{P}_a$. We use the same loss as [25] that disentangles rotation, depth and image-plane translations [48] (see equations in the appendix [26]). For $\mathcal{L}_q$, we use a simple L_2 regression loss, $\mathcal{L}_q = \|q^k + \Delta q^k - q^{gt}\|^2_2$.

Note that the 6D pose loss is measured only on the anchor part a while the alignment of the other parts of the robot is measured by the error on their joint angles (rather than alignment of their 3D point clouds). This disentangles the 6D pose loss $\mathcal{L}_a$ from the joint angle loss $\mathcal{L}_q$ and we found this leads to better convergence. We sum the loss over the refinement iterations k to imitate how the refinement algorithm is applied at test time but the error gradients are not backpropagated through rendering and iterations. Finally, for simplicity the loss (3) is written for a single training example, but we sum it over all examples in the training set.

Training data. For training the refiner, we use existing datasets [28, 61] provided by prior works for the Kuka, Panda, Baxter, OWI-535 robots. All of these datasets are synthetic, generated using similar procedures based on domain randomization [51, 32, 47, 20]. The joint angles are sampled independently and uniformly within their bounds, without assuming further knowledge about their test time distribution. We add data augmentation similar to [25].

We sample the initial state $\mathcal{S}^0$ by adding noise to the ground truth state in order to simulate errors of the network prediction at the previous state of the refinement as well as the error at the initalization. For the pose, we sample a translation from a centered Gaussian distribution with standard deviation of 10 cm, and a rotation by sampling three angles from a centered Gaussian distribution of standard deviation $60°$. For the joint angles, we sample an additive noise from a centered Gaussian distribution with a standard deviation equal to 5% of the joint range of motion, which is around $20°$ for most of the joints of the robots we considered.

Implementation details. We train separate networks for each robot. We use a standard ResNet-34 architecture [12] as the backbone of the deep refiner. The hyper-parameters are $\lambda = 1$ and $K = 3$ training iterations. Note that at test time we can perform more iterations, and the results we report correspond to 10 iterations. The anchor is sampled randomly among the 5 largest parts of the robot at each iteration. This choice is motivated in Section 3.4 and other choices are considered in the experiments, Section 4.3. We initialize the network parameters randomly and perform the optimization using Adam [24], with the procedure described in the appendix [26] for all the networks.

Known joint angles at test time. The approach described previously could be used at test time with measured joint angles $q^0 = q^{gt}$ and by ignoring the joint update, but we observed better results by training a separate network which only predicts a pose update for this scenario. In this context where the joint values are known and constant, the full robot is considered as a single and unique anchor. Yet, the problem remains different from classic rigid object 6D object pose estimation because the network must generalize to new joint configurations unseen during training.

3.4. Parametrization choices

There are two main parametrization choices in our approach: (i) the choice of the reference point O for the parametrization of the pose update $\Delta\mathcal{T}^k$ in equation (2) and (ii) the choice of the anchor part to update the 6D pose and measure pose loss in equation (3). These choices have a significant impact on the results, as shown in Section 4.

Choice of the reference point for the pose update. Similar to [31], we parametrize the pose update as a rotation around a reference point O and a translation defined as a function of the position of O with respect to the camera. The fact that the rotation is around O is a first obvious influence of this choice on the transformation that needs to be predicted. The impact on the translation update parameters is more complicated: they are defined by a multiplicative update on the depth of O and by an equivalent update in pixels in the image, which is also related to the real update by the depth of O (see equations in the appendix [26]).

A seemingly natural choice for reference point O would be a physical point on the robot, for example the center of the base of the robot or the anchor part. However, on the contrary to the rigid object case, if that part is not visible or is partially occluded, the network cannot infer the position of the reference O precisely, and thus cannot predict a relevant translation and rotation update. In experiments, we show it is better to use as O the centroid of the estimated robot state, which takes into account the estimated joint configuration, and can be more reliably estimated.

Choice of the anchor part. The impact of the choice of the anchor part $\mathcal{P}_a$ used for computing the 6D pose loss in equation (3), is illustrated in Figure 4. We explore several choices of anchor part in our experiments, and show that this choice has a significant impact on the results. Since the optimal choice depends on the robot, and the observed pose, we introduce a strategy where we randomly select the anchor among the largest parts of the robot, during both training and refinement, and show that on average it performs similarly or slightly better than the optimal oracle choice of a single unique anchor on the test set.

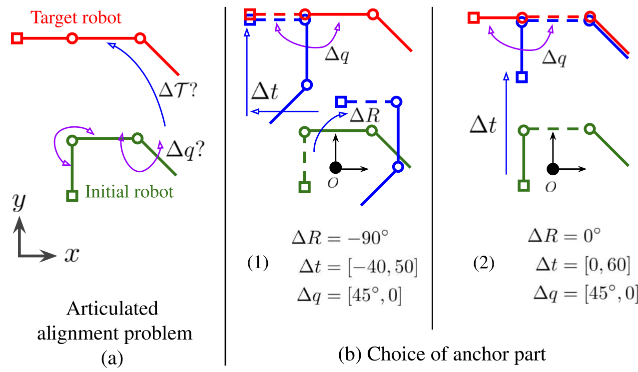


Figure 4: **Choice of the anchor part. (a)** We analyze how the choice of the anchor part affects the complexity of the rigid alignment $\Delta\mathcal{T}$ and the joint angle update Δq to align an initial state of the robot (green) with the target state of the robot (red). **(b)** We show the required rigid pose update (composed of a rotation and a translation) and the required joint update for two different choices of the anchor part (shown using a dashed line). In (1), the required pose update of the anchor part consists of successively applying rotation ΔR and translation Δt along x and y axes (in blue). In (2), the anchor part is aligned using only a translation along the y axis resulting in a simpler solution compared to (1). These examples illustrate that the choice of the anchor can have a significant impact on the complexity of the alignment problem.

4. Experiments

We evaluate our method on recent benchmarks for the following two tasks: (i) camera-to-robot 6D pose estimation for three widely used manipulators (Kuka iiwa7, Rethink robotic Baxter, Franka Emika Panda) [28], and (ii) full state estimation of the low-cost 4 DoF robotic arm OWI-535 [61]. In Section 4.1, we consider the first task, where an image of a robot with fixed known joint angles is used to estimate the 6D camera-to-robot pose. We show that our approach outperforms the state-of-the-art DREAM method [28]. In Section 4.2, we evaluate our full approach where both the 6D pose and joint angles are unknown. We show our method outperforms the state-of-the-art method [61] for this problem on their dataset depicting the low-cost 4 DoF robot and that it can recover the 6D pose *and* joint angles of more complex robotic manipulators. Finally, Section 4.3 analyzes the parametrization choices discussed in Section 3.4.

4.1. 6D pose estimation with known joint angles

Datasets and metrics. We focus on the datasets annotated with 6D pose and joint angle measurements recently introduced by the state-of-the-art method for single-view camera-to-robot calibration, DREAM [28]. We use the provided training datasets with 100k images generated with domain

Robot	Robot (DoF)	Real	# images	# 6D poses	cam.	DREAM [28] VGG19-F	DREAM [28] VGG19-Q	DREAM [28] ResNet101-H	Ours ResNet34	Ours Unknown angles
	Dataset informations									
Baxter DR	Baxter (15)	×	5982	5982	GL	-	75.47	-	**86.60**	**32.35**
Kuka DR	Kuka (7)	×	5997	5997	GL	-	-	73.30	**89.62**	**80.44**
Kuka Photo	Kuka (7)	×	5999	5999	GL	-	-	72.14	**86.87**	**78.62**
Panda DR	Panda (8)	×	5998	5998	GL	81.33	77.82	82.89	**92.70**	**82.19**
Panda Photo	Panda (8)	×	5997	5997	GL	79.53	74.30	81.09	**89.89**	**77.15**
Panda 3CAM-AK	Panda (8)	✓	6394	1	AK	68.91	52.38	60.52	**76.53**	**66.86**
Panda 3CAM-XK	Panda (8)	✓	4966	1	XK	24.36	37.47	64.01	**85.97**	**78.66**
Panda 3CAM-RS	Panda (8)	✓	5944	1	RS	76.13	77.98	**78.83**	76.90	**77.06**
Panda ORB	Panda (8)	✓	32315	27	RS	61.93	57.09	69.05	**80.54**	**68.87**

Table 1: Comparison of RoboPose (ours) with the state-of-the-art approach DREAM [28] for the camera-to-robot 6D pose estimation task using the 3D reconstruction ADD metric (higher is better). The robot joint configuration is assumed to be known (results in black) and is different in each of the image in the dataset, but the pose of the camera with respect to the robot can be fixed (# number of 6D poses). Multiple cameras are considered to capture the input RGB images: synthetic rendering (GL), and real Microsoft Azure (AK), Microsoft Kinect360 (XK) and Intel RealSense (RS), which all have different intrinsic parameters. Our results in blue do not use ground truth joint angles (see Section 4.2) and the accuracy of the robot 3D reconstruction is evaluated using both the estimated 6D pose and the joint angles.

randomization. Test splits are available as well as photorealistic synthetic test images (Photo). For the Panda robot, real datasets are also available. The Panda 3CAM datasets display the fixed robot performing various motions captured by 3 fixed different cameras with different focal lengths and resolution, all of which are different than the focal length used during training. The largest dataset with the more varied viewpoints is Panda-ORB with 32,315 real images in a kitchen environnement captured from 27 viewpoints with different joint angles in each image.

We use the 3D reconstruction ADD metric which directly measures the pose estimation accuracy, comparing distances between 3D keypoints defined at joint locations of the robot in the ground truth and predicted pose. We refer to the appendix [26] for exact details on the evaluation protocol of our comparison with DREAM [28].

Comparison with DREAM [28]. We train one network for each robot using the same synthetic datasets as [28] and report our results in Table 1. Our method achieves significant improvements across datasets and robots except on Panda 3CAM-RS where the performance of [28] with ResNet101-H variant is similar to ours. On the Panda 3CAM-AK and Panda 3CAM-XK datasets, the performance of our method is significantly higher than the ResNet101-H model of [28] (e.g. +21.96 on 3CAM-XK), which suggests that the approach of [28] based on 2D keypoints is more sensitive to some viewpoints or camera parameters. Note that our method trained with the synthetic GL camera can be applied to different real cameras with different intrinsics at test time thanks to our cropping strategy which provides an augmentation of the effective focal length during training.

On Panda-ORB, the largest real dataset that covers multiple camera viewpoints, our method achieves a large improvement of 11.5 points. Our performance on the synthetic datasets for the Kuka and Baxter robots is also significantly higher than [28]. We believe the main reason for this large improvements is the fact that our Render & Compare approach can directly use the shape of the entire robot rendered in the observed configuration for estimating the pose rather than detecting a small number of keypoints.

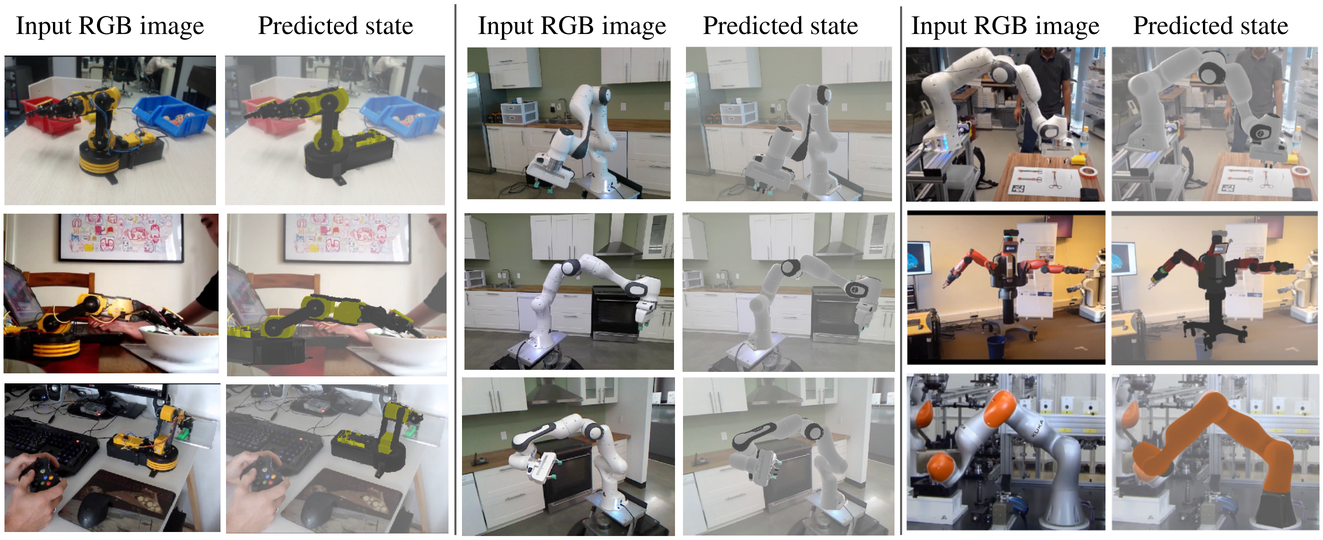
	Ours (individual frames) K=1	K=2	K=3	K=5	K=10	Ours (online) K=1	DREAM [28] ResNet101-H
ADD	28.5	72.8	79.1	80.4	**80.7**	80.6	69.1
FPS	16	8	4	2	1	16	**30**

Table 2: Benefits of iterative refinement and running time on Panda-ORB video sequence of robot trajectories. We report ADD and running time (frames per second, FPS) for a varying number of refiner iterations K. The frames are either considered individually, or the estimate is used to initialize the refiner in the subsequent frames (online) without additionnal temporal filtering.

Running time. In Table 2 we report the running time of our method on the Panda-ORB dataset which consists of robot motion videos captured from 27 different viewpoints. The first observation is that the accuracy increases with the number of refinement iterations K used at test-time, and the most significant improvements are during the first 3 iterations. The importance of using multiple network iterations during training is further discussed in the appendix [26]. We also report an online version of our approach that leverages temporal continuity. It runs the refiner with $K = 10$ iterations on the first frame of the video and then uses the output pose as the initialization for the next frame and so on, without any additional temporal filtering of the resulting 6D poses. This version runs at 16 frames per second (FPS) and achieves a similar performance as the full approach that considers each frame independently and runs at 1 FPS.

4.2. 6D pose and joint angle estimation

We now evaluate the performance of our method in the more challenging scenario where the robot joint configuration angles are unknown and need to be estimated jointly with the 6D pose



(a) (b) (c)

Figure 5: Qualitative results of RoboPose 6D pose and joint angle estimation for four different robots. (a) The OWI-535 robot from the CRAVES-lab (first row) and CRAVES-youtube (second and third row) datasets, (b) the Panda robot from the Panda 3CAM dataset and (c) the Panda, Baxter and Kuka robots on example images from the Internet. **Please see additional results in the appendix [26].**

	CRAVES [61] synt	CRAVES [61] synt+real*		ours synt	
PCK@0.2	95.66	**99.55**		99.28	
Error		all	top 50%	all	top 50%
Joints (degrees)		11.3	4.74	**5.44**	**3.34**
Trans xyz. (cm)		10.1	5.52	**0.63**	**0.49**
Trans norm. (cm)		19.6	10.5	**1.34**	**1.01**
Rot. (degrees)		10.3	5.29	**4.36**	**3.19**

Table 3: Results on the CRAVES-lab [61] dataset with unknown joint angles. We report average errors on *all* the images of the dataset, or on the *top 50%* images selected according to the best joint angle accuracy with respect to the ground truth. Networks are trained on synthetic data only (synt) or also using non-annotated real images of the robot (synt+real*).

from a single RGB image. Qualitative results on the considered datasets as well as on real images crawled from the web are shown in Figure 5. Please see the appendix [26] for additional qualitative examples, and the project page [1] for a movie showing our predictions on several videos.

Comparison with CRAVES [61]. CRAVES [61] is the state-of-the-art approach for this task. We consider the two datasets used in [61]. CRAVES-lab displays the OWI-535 4DoF in a lab environment and contains 20,000 RGB images of which 428 key frames are annotated with 2D robot keypoints, ground truth joint angles (not used by our method) and camera intrinsics. CRAVES-youtube is the second dataset containing real-world images crawled from YouTube depicting large variations in viewpoint, illumination conditions and robot appearance. It contains 275 frames annotated with 2D keypoints but no camera intrinsic parameters, 6D pose or joint angle ground truth. In addition to

CRAVES synt [61]	CRAVES synt+real* [61]	Ours, synt f=500	f=1000	f=1500	f=2000	f=best
81.61	**88.89**	85.16	86.96	84.87	83.57	91.48

Table 4: PCK@0.2 on the CRAVES-Youtube dataset [61].

metrics that measure the 6D pose and joint angle estimates, we report a 2D keypoint metric, PCK (percentage of keypoints), following [61]. We refer to the appendix [26] for details of the metrics and the evaluation protocol.

We compare with two variants of CRAVES, one trained only on synthetic images (synt), and one that also requires real non-annotated images (synt+real*). Our method is trained only using the $5,000$ provided synthetic images. We report results on CRAVES-lab in Table 3. To compare with the 2D keypoint metric PCK@0.2, we project in the image the 3D keypoints of our estimated robot state. On this metric, our method outperforms CRAVES trained only on synthetic images and achieves a near-perfect score, similar to their approach trained with real images. More importantly, we achieve much better results when comparing with the 3D metrics (joint angles error and translation/rotation error). CRAVES achieves high average errors when all images of the datasets are considered, which is due to the complexity of solving the nonlinear nonconvex 2D-to-3D optimization problem for recovering 6D pose and joint angles given 2D keypoint positions. Our method trained to directly predict the 6D pose and joint angles, achieves large improvements in precision. We reduce the translation error by a factor of 10, demonstrating robustness to depth ambiguities.

We also evaluated our method on CRAVES-youtube. On this dataset, the camera intrinisic parameters are unknown and cannot be used for projecting the estimated robot pose

Reference point	ADD
on Root $\mathcal{P}_0$	75.02
on Middle $\mathcal{P}_4$	79.45
on Hand $\mathcal{P}_7$	00.00
Centroid (ours)	**80.54**

(a)

Reference point	Volume (cm^3)	ADD
on $\mathcal{P}_5$	3092	74.40
on $\mathcal{P}_2$	2812	75.06
on $\mathcal{P}_1$	2763	74.89
on $\mathcal{P}_0$	2660	75.02
on $\mathcal{P}_4$	2198	79.45
on $\mathcal{P}_7$	637	0.00
Centroid (ours)	-	**80.54**

(b)

Table 5: **Analysis of the choice of reference point O.** Networks are trained and evaluated with known joint angles as in Section 4.1. The reference point is placed on (a) a naively chosen part and (b) on one of the 5 largest parts. Our strategy of using the centroid of the imaged robot performs the best.

into the 2D image. We therefore report results for different hypothesis of (fixed) focal lengths for all the images of the dataset, as well as using an oracle (f=best) which selects the best focal length for each image. Results are reported in Table 4. For 2D keypoints, our method for $f = 1000$ achieves results superior to CRAVES trained only on synthetic images, and also outperforms CRAVES trained with real data when selecting the best focal length. 3D ground truth is not available, but similar to CRAVES-lab we could expect large improvements in 3D accuracy.

Experiments on 7DoF+ robots. We also train our method for jointly predicting 6D pose and joint angles for the robots considered in Section 4.1. We evaluate the 6D pose *and* joint angles accuracy using ADD. Results are reported in Table 1 in blue (last column). For the 7DoF robotic arms (Kuka and Panda), these results demonstrate a competitive or superior ADD accuracy compared to [28] for inferring the 3D geometry of a known robot, but our method does not require known joint angles. The more complex 15 DoF Baxter robot remains challenging although our qualitative results often show reasonable alignments. We discuss the failure modes of our approach in the appendix [26].

4.3. Analysis of parametrization choices

We analyze our method on the Panda-ORB dataset: it is the largest real dataset containing significant variations in joint angles and camera viewpoints and the Panda robot has a long kinematic chain with 8 DoF. We study the choice of reference point O for the 6D pose update and the choice of the anchor part (see Section 3.4).

Reference point. We train different networks with the reference point at the origin of the root $\mathcal{P}_0$, the part in the middle of the kinematic chain $\mathcal{P}_4$ and at the end of the kinematic chain $\mathcal{P}_7$. Results are reported in Table 5(a). We observe that the performance indeed depends on the choice of the reference point. The network trained with the “Hand” part ($\mathcal{P}_7$, the end effector) as a reference point fails to converge because this part is often difficult to identify in the training images and its pose cannot be inferred from any other part because the robot is not a rigid object. We investigate picking the reference point on one of the five largest parts (measured by their 3D volume which is correlated with 2D visibility) in Table 5(b) again demonstrating our approach of using the centroid of the robot performs better than any of these specific parts.

Anchor	Volume (cm^3)	ADD
$\mathcal{P}_5$	3092	68.01
$\mathcal{P}_2$	2812	65.56
$\mathcal{P}_1$	2763	60.40
$\mathcal{P}_0$	2660	57.44
$\mathcal{P}_4$	2198	**69.54**
$\mathcal{P}_7$	637	63.40

(a)

Anchor	ADD
Root part $\mathcal{P}_0$	57.44
Middle part $\mathcal{P}_4$	69.54
Hand part $\mathcal{P}_7$	63.40
Random (all)	64.24
Random (5 largest)	68.87
Random (3 largest)	**71.33**

(b)

Table 6: **Analysis of the choice of the anchor part.** Networks are trained and evaluated with unknown joint angles as in Section 4.2. (a) Results when one fixed anchor part is used during training and testing. (b) Randomly selecting the anchor part among a given set of largest robot parts during refinement in both training and testing.

Choice of the anchor part. Table 6 reports results using different strategies for chosing the anchor part during training and testing. First, in 6(a) we show that choosing different parts as one (fixed) anchor results in significant variation in the resulting performance. To mitigate this issue we consider in 6(b) a strategy where the anchor is picked randomly among the robot parts at each iteration (both during training and testing). This strategy performs better than always naively selecting the root $\mathcal{P}_0$ as anchor. By restricting the sampled anchors to the largest parts, our automatic strategy can also perform better than the best performing part $\mathcal{P}_4$.

5. Conclusion

We have introduced a new *render & compare* approach to estimate the joint angles and the 6D camera-to-robot pose of an articulated robot from a single image demonstrating significant improvements over prior state-of-the-art for this problem. These results open-up exciting applications in visually guided manipulation or collaborative robotics without fiducial markers or time-consuming hand-eye calibration. To stimulate these applications, we released the training code as well as the pre-trained models for commonly used robots.

Acknowledgments. This work was partially supported by the HPC resources from GENCI-IDRIS (Grant 011011181R1), the European Regional Development Fund under the project IMPACT (reg. no. CZ.02.1.01/0.0/0.0/15 003/0000468), Louis Vuitton ENS Chair on Artificial Intelligence, and the French government under management of Agence Nationale de la Recherche as part of the ”Investissements d’avenir” program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute).

References

[1] Project page: https://www.di.ens.fr/willow/research/robopose/. 1, 7

[2] Ben Abbatematteo, Stefanie Tellex, and George Konidaris. Learning to generalize kinematic models to novel objects. In *CoRL*, 2019. 2

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006. 2

[4] Jeannette Bohg, Javier Romero, Alexander Herzog, and Stefan Schaal. Robot arm pose estimation through pixel-wise part classification. In *ICRA*, 2014. 2

[5] Fabrizio Caccavale and Masaru Uchiyama. Cooperative manipulation. In *Handbook of robotics*. Springer, 2016. 1

[6] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *The international journal of robotics research*, 2011. 2

[7] A Collet and S S Srinivasa. Efficient multi-view object recognition and full pose estimation. In *ICRA*, 2010. 2

[8] Karthik Desingh, Shiyang Lu, Anthony Opipari, and Odest Chadwicke Jenkins. Factored pose estimation of articulated objects using efficient nonparametric belief propagation. In *ICRA*, 2019. 2

[9] Mark Fiala. Artag, a fiducial marker system using digital techniques. In *CVPR*, 2005. 2

[10] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 2014. 2

[11] Karol Hausman, Scott Niekum, Sarah Osentoski, and Gaurav S Sukhatme. Active articulation model estimation through interactive perception. In *ICRA*, 2015. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[13] J Heller et al. Structure-from-motion based hand-eye calibration using L-∞ minimization. In *CVPR*, 2011. 2

[14] S Hinterstoisser, S Holzer, C Cagniart, S Ilic, K Konolige, N Navab, and V Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *ICCV*, 2011. 2

[15] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders GlentBuch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, et al. Bop: Benchmark for 6d object pose estimation. In *ECCV*, 2018. 2

[16] Tomáš Hodaň, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas. BOP challenge 2020 on 6D object localization. In *ECCVW*, 2020. 2

[17] Radu Horaud and Fadi Dornaika. Hand-eye calibration. *The international journal of robotics research*, 1995. 2

[18] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6d object pose estimation. In *CVPR*, 2019. 2

[19] Jarmo Ilonen and Ville Kyrki. Robust robot-camera calibration. In *ICAR*, 2011. 2

[20] Stephen James and Paul et al. Wohlhart. Sim-to-Real via Sim-to-Sim: Data-efficient robotic grasping via Randomized-to-Canonical adaptation networks. *CVPR*, 2019. 4

[21] Dov Katz and Oliver Brock. Manipulating articulated objects with interactive perception. In *ICRA*, 2008. 2

[22] Dov Katz, Moslem Kazemi, J Andrew Bagnell, and Anthony Stentz. Interactive segmentation, tracking, and kinematic modeling of unknown 3d articulated objects. In *ICRA*, 2013. 2

[23] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *ICCV*, 2017. 2

[24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[25] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *ECCV*, 2020. 2, 3, 4

[26] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. Single-view robot pose and joint angle estimation via render & compare. *Extended version on arXiv*, 2021. 3, 4, 5, 6, 7, 8

[27] Jens Lambrecht and Linh Kästner. Towards the usage of synthetic data for marker-less pose estimation of articulated robots in rgb images. In *ICAR*, 2019. 2

[28] Timothy E Lee, Jonathan Tremblay, Thang To, Jia Cheng, Terry Mosier, Oliver Kroemer, Dieter Fox, and Stan Birchfield. Camera-to-robot pose estimation from a single image. In *ICRA*, 2020. 1, 2, 4, 5, 6, 8

[29] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *IJCV*, 2009. 2

[30] Xiaolong Li, He Wang, Li Yi, Leonidas J Guibas, A Lynn Abbott, and Shuran Song. Category-level articulated object pose estimation. In *CVPR*, 2020. 2

[31] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *ECCV*, 2018. 2, 3, 4, 5

[32] Vianney Loing, Renaud Marlet, and Mathieu Aubry. Virtual training for a real application: Accurate object-robot relative localization without calibration. *IJCV*, 2018. 4

[33] David G Lowe. Three-dimensional object recognition from single two-dimensional images. *Artif. Intell.*, 1987. 2

[34] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999. 2

[35] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In *ECCV*, 2018. 2

[36] Roberto Martín-Martín, Sebastian Höfer, and Oliver Brock. An integrated approach to visual perception of articulated objects. In *ICRA*, 2016. 2

[37] Frank Michel, Alexander Krull, Eric Brachmann, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. Pose estimation of kinematic chain instances via object coordinate regression. In *BMVC*, 2015. 2

[38] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Generalized feedback loop for joint hand-object pose estimation. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 2

[39] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *ICRA*, 2011. 2

[40] Frank C Park and Bryan J Martin. Robot sensor calibration: solving ax= xb on the euclidean group. *IEEE Transactions on Robotics and Automation*, 1994. 2

[41] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *ICCV*, 2019. 2

[42] Karl Pauwels, Leonardo Rubio, and Eduardo Ros. Real-time model-based articulated object pose detection and tracking with variable rigidity constraints. In *CVPR*, 2014. 2

[43] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *ICRA*, 2017. 2

[44] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019. 2

[45] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *CVPR*, 2017. 2

[46] Lawrence G Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963. 2

[47] Fereshteh Sadeghi, Alexander Toshev, Eric Jang, and Sergey Levine. Sim2real viewpoint invariant visual servoing by recurrent control. In *CVPR*, 2018. 4

[48] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *ICCV*, 2019. 4

[49] Chen Song, Jiaru Song, and Qixing Huang. Hybridpose: 6d object pose estimation under hybrid representations. In *CVPR*, 2020. 2

[50] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *CVPR*, 2018. 2

[51] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, 2017. 4

[52] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stan Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *CoRL*, 2018. 2

[53] Jonathan Tremblay, Stephen Tyree, Terry Mosier, and Stan Birchfield. Indirect object-to-robot pose estimation from an external monocular rgb camera. *IROS*, 2020. 2

[54] Roger Y Tsai, Reimar K Lenz, et al. A new technique for fully autonomous and efficient 3 d robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, 1989. 2

[55] Felix Widmaier, Daniel Kappler, Stefan Schaal, and Jeannette Bohg. Robot arm pose estimation by pixel-wise regression of joint angles. In *ICRA*. 2

[56] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In *RSS*, 2018. 2

[57] Dekun Yang and John Illingworth. Calibrating a robot camera. In *BMVC*, 1994. 2

[58] Li Yi, Haibin Huang, Difan Liu, Evangelos Kalogerakis, Hao Su, and Leonidas Guibas. Deep part induction from articulated object pairs. *arXiv preprint arXiv:1809.07417*, 2018. 2

[59] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *CVPR*, 2019. 2, 3

[60] Tao Zhou and Bertram E Shi. Simultaneous learning of the structure and kinematic model of an articulated body from point clouds. In *International Joint Conference on Neural Networks (IJCNN)*, 2016. 2

[61] Yiming Zuo, Weichao Qiu, Lingxi Xie, Fangwei Zhong, Yizhou Wang, and Alan L Yuille. Craves: Controlling robotic arm with a vision-based economic system. In *CVPR*, 2019. 1, 2, 4, 5, 7