# I Like to Move It: 6D Pose Estimation as an Action Decision Process

Benjamin Busam, Hyun Jun Jung, and Nassir Navab

Technical University of Munich
b.busam@tum.de hyunjun.jung@tum.de navab@cs.tum.edu

**Abstract.** Object pose estimation is an integral part of robot vision and augmented reality. Robust and accurate pose prediction of both object rotation and translation is a crucial element to enable precise and safe human-machine interactions and to allow visualization in mixed reality. Previous 6D pose estimation methods treat the problem either as a regression task or discretize the pose space to classify. We reformulate the problem as an action decision process where an initial pose is updated in incremental discrete steps that sequentially move a virtual 3D rendering towards the correct solution. A neural network estimates likely moves from a single RGB image iteratively and determines so an acceptable final pose. In comparison to previous approaches that learn an object-specific pose embedding, a decision process allows for a lightweight architecture while it naturally generalizes to unseen objects. Moreover, the coherent action for process termination enables dynamic reduction of the computation cost if there are insignificant changes in a video sequence. While other methods only provide a static inference time, we can thereby automatically increase the runtime depending on the object motion. We evaluate robustness and accuracy of our action decision network on video scenes with known and unknown objects and show how this can improve the state-of-the-art on YCB videos [81] significantly.

**Keywords:** 3D Object Detection, 6D Pose Estimation, Synthetic Data

## 1 Introduction

We live in a 3D world. Every object with which we interact has six degrees of freedom to move freely in space, three for its orientation and three for its translation. Thus, the question to determine these parameters naturally arises whenever we include a vision system observing the scene. A single camera will only observe a projection of this world. Thus, recovering such 3D information constitutes an inherently ill-posed problem which has drawn attention of many vision experts in the past [34,41,25,51,81]. The motives for this can be different: One may want to extract scene content for accurate measurements [4], camera localization [52] or 3D reconstruction [40]. Another driver can be geometric image manipulation [29,8] or sensor fusion [17]. Also human-robot interaction [6]

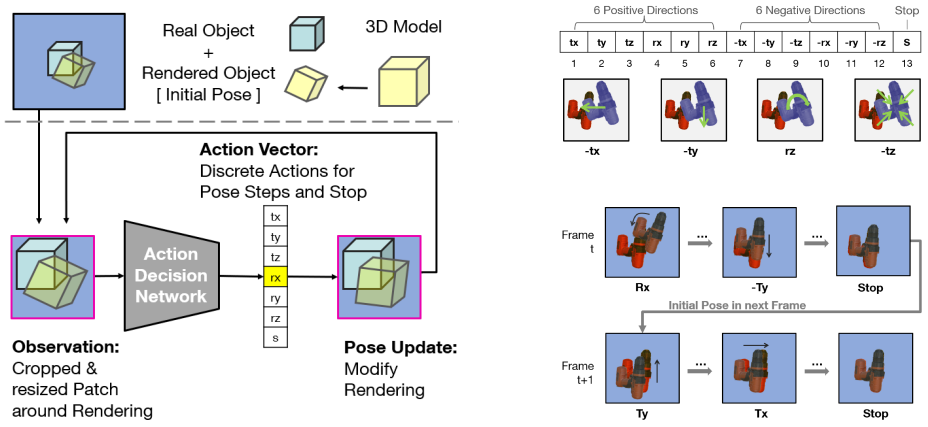


Fig. 1: **Method Overview.** Left: To estimate the pose of the real object, a virtual object is rendered with an initial pose (top left). Both image and rendering are cropped (RoI in pink). A lightweight action decision network determines an incremental move to bring the rendering closer to the real observation. The updated pose is used to iteratively modify the rendering. **Actions Decision Process.** Right: There are 13 possible actions: 6 pose actions to move in positive, 6 to move in negative direction, and one action for stop. An example is shown where the rendering is moved until stop is predicted. This pose initializes the next frame.

and robot grasping [16] require estimation of 6D poses.
The rise of low-cost RGBD sensors helped development of 6D pose detectors [5,36,76] and tracker [70,22]. More recently, the field also considered methods with single RGB image input. The best performing methods for this task [56,84,27] are all data-driven and thus require a certain amount of training images. Annotating a large body of data for this kind of task is cumbersome and time-intensive which yields to either complex acquisition setups [22] or diverse annotation quality as we will discuss in Section 4. Oftentimes, high quality 3D models of the objects exist a priori [81,22]. The majority of pose estimation pipelines (e.g. [72,81,56]) are trained on real data. Besides difficult and time-consuming annotations, this brings two further drawbacks. On one hand, the networks adjust to the individual sensor noise of the acquisition hardware drastically hampering generalization capabilities [33]. On the other hand, every real annotation has its own errors introduced either by the used ground truth sensor system or by the human annotator, thus propagating this error to every model trained on it. Modern 3D renderers, however, can produce photorealistic images in high quantity with pixel-perfect ground truth. Some recent scholars therefore propose to leverage such data [35,69,84] and fully train on synthetic images. Most widely used evaluation datasets [25,5] provide single image acquisitions and only recently video sequences [81,22] with pose annotations are available even though video data is

the natural data source in applications.

**Contributions and Outline.** We leverage the temporal component in video data to accelerate our pose estimation performance and propose an RGB pose estimation pipeline by taking inspiration from the reinforcement learning approach proposed for 2D bounding box tracking [83] where the authors frame the problem with consecutive discrete actions for an agent. We frame 6D pose estimation as an action decision process realized by applying a network that determines a sequence of likely object move as shown in Fig. 1. At first, an initial pose is used to render the 3D model. Both the rendering and the current image are cropped around the virtual pose and fed to a lightweight CNN. The network predicts a pose action to move the rendering to closer to the real object. The stepsize is hereby fixed and predefined. It determines the accuracy of the process and the convergence speed. In case the process continues, the pose is modified according to the action and the new rendering is fed back into the pipeline with a new crop to move the estimation incrementally closer to the observation. This goes on until either the stop criterion fires or the maximum number of iterations is reached. If our input is a video stream, we can use the pose retrieved at frame $t$ as an initial pose for frame $t + 1$ which can greatly reduce the computation time as the amount of iterations is determined by the pose actions needed between the initial pose and the result. Improving pose estimation with iterative inference has previously been explored by [43] where a refinement network is iteratively applied to refine a pose predicted by an estimator such as PoseCNN [81]. However, the performance of their method actually decreases if more than two iterations are used.
In summary, our contributions in this work are fourfold:

1. We reformulate 6D pose estimation as an **action decision process** and design a lightweight CNN architecture for this task that **generalizes to unseen objects**.
2. We **iteratively** apply our shallow network to optimize the pose and deploy a change-aware **dynamic complexity reduction** scheme to improve inference cost.
3. We provide an RGB-only method that is able to **improve** the state-of-the-art for **video pose estimation** while being able to track objects in presence of noise and clutter.
4. We provide a **data augmentation scheme** to render high-quality images of 3D models on real backgrounds under varying clutter and occlusion.

In the remainder of the paper, we first review the related literature in Section 2 before we present our method and network architecture in detail (Sec. 3). We report an extensive analysis and evaluation of our method in Sec. 4 and give some retrospect in Sec. 5.

## 2 Related Work

Vision system acquire imagery of our 3D world. In order to interact with objects in this world it is crucial to understand relative position and orientation. As this process is immanent for every real 3D camera application, many different solutions have been proposed to estimate rigid object and camera poses.

**From Markers to Features.** Early works apply marker based systems to track objects. Typical augmented reality applications are driven by markers such as AR-Tag [19], ArUcO [23], ARToolkit [34] or AprilTag [54]. These are also used for sensor fusion [17] and extended to high accuracy systems [4]. Reliable and robust detection is of particular interest in the medical domain [18], where self-adhesive markers allow flexible usage [6].

Object-marker calibration can be intricate and time-consuming in practice and feature extractors are a practicable alternative. Methods such as SIFT [47], SURF [2], BRISK [42], ORB [65] etc. are utilized for camera [51,52] and object [80,44] pose estimation. Tracking applications benefit from the rotation accuracy of such systems in inside-out camera setups [9]. The Perspective-$n$-Point (P$n$P) algorithm and its successor EP$n$P [41] are still utilized to recover 6D poses from 2D-3D correspondences. The rise of modern RGB-D sensors also triggered the design of 3D descriptors [67,73] for accurate object retrieval even in cluttered scenes [50].

**Pose Regression and Classification.** Rotations densely populate a non-Euclidean space and there are multiple parametrization for the Riemannian manifold described by them [7]. On the unit quaternions hypersphere for instance, the geodesic distance is not compliant with the Euclidean L-p norm in its 4D-embedding and the parametrization constitutes a double cover of the rotation group $SO(3)$ impeding 6D pose regression networks [86]. Some works therefore discretize the problem and classify [35]. Hinterstoisser [25] uses a template matching strategy for viewpoint estimation and [10,38,28] achieve a sub-linear matching complexity in the number of objects by hashing.

Others train a regressor for RGBD [5,71,79,36,76] pose estimation. Some scholars have recently also reported methods that solely rely on RGB [12,35,59,14,81,69] input without the need of additional depth.

To realize a 6D pose estimation pipeline, these methods are usually separated into three stages [35,69,59]: 2D detection, 2D keypoint extraction, 6D pose estimation. Tekin [72] is based on YOLO [62] and thus provides a single shot method. After bounding box corner or keypoint detection, the 6D pose is estimated with P$n$P. Other approaches [30,81,14,55] utilize multi-modalities or multi-task training. More recently, pixel-wise object correspondences [84,56,45] use robust P$n$P within a RANSAC loop to improve the results. The model performance is mostly hampered by the domain gap created through synthetic-only data training which is addressed for depth renderings by [60] which extends the ideas of [61]. Further works address occlusion [53,20] and ambiguous [48] cases. To improve upon the estimated pose, Li et al. [43] propose an RGB-based refinement strategy. Many methods, however, refine their RGB results with additional depth information using ICP [85]. All the core networks usually require to train one network per ob-

ject. If training is is done for multiple objects, the resulting predictions become unreliable [33]. The recent CorNet [58] focuses on objects geometry instead and detects object-agnostic corners. While this is more robust, it is in spirit similar to early pose estimation approaches that detect significant points. Our model is different as we learn a discrete set of decisions that lead to the correct pose. This provides the flexibility of object-specific training as well as object agnostic decisions trained with a heterogeneous dataset.

**Temporal Tracking.** Tracking of 3D objects using temporal information has been presented with the help of depth maps and point clouds. It can be realized with ICP [3] and its variants [66,68]. These methods highly rely on an initial pose close to the correct prediction and fail in the presence of heavy noise and clutter [22]. To stabilise tracking with depth maps, additional intensity information [24,82,31,37] or a robust learning procedure [70] helps. The current methods need one CNN trained per objects [21] or are bound to specific geometrical constraints such as planar objects [77]. More recently, PoseRBPF [13] was presented as an efficient RGB-only tracker using a particle filter setting state-of-the-art results on the YCB dataset [81]. Although our approach may appear similar to a classical temporal tracker whose optimization procedure usually includes incremental pose updates and requires initialization close to the correct pose in order not to fail or drift [85], the convergence basin of our method is much wider (see Sec. 4.3). While we largely benefit from temporal information in terms of computation time, our method can also be used to detect the pose with multiple seeds intuitively (see Sec. 4.4).

**Pose Datasets.** To compare different tracking and detection method for 6D pose estimation, different datasets have been proposed. LineMOD [25] and its occlusion extension [5] are arguably the most widely used ones for detection. More recently HomebrewedDB [33] uses three of the LineMOD objects and adds 30 higher quality 3D models. The scenes are more cluttered and acquired under different illumination conditions. Other datasets focus on textureless industrial [26,16] and household [63,15,71] objects. These datasets together with several others are summarized in the BOP 6D Pose Benchmark [27]. While the different setups are diverse and the ground truth labels often of very high quality, objects are usually acquired from individual acquisitions that are not temporally connected making tracking evaluation difficult. The more recent YCB-Video dataset presented by [81], however, includes 92 video sequences of 21 household objects and 12 test videos with annotated ground truth poses and detailed 3D models. Several RGBD trackers also evaluate on the dataset of Garon et al. [22] that includes severe occlusion and clutter.

## 3 Method

Our target is to optimize an action decision CNN to decide for iterative discrete actions to move a rendered 3D model to the observed position of the according object in an image sequence as shown in Fig. 1. An initial pose is used to crop the image with the projected bounding box of the object. We discretize the set of

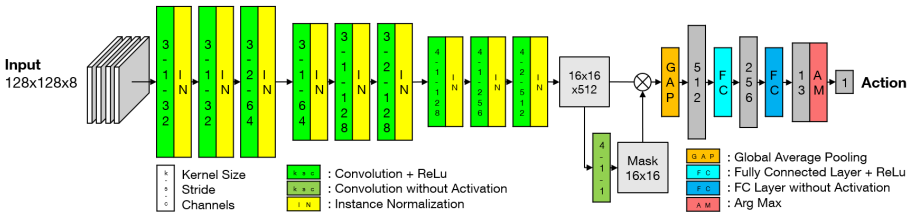


Fig. 2: **Architecture Overview.** The input is the RGB video frame cropped and concatenated with the rendered RGB, rendered depth and rendered segmentation mask. A series of convolutional layers with activations are used to extract an embedding. An unsupervised attention mask is concatenated with it before an global average pooling layer. Two fully connected layers extract the set of action logits from which the most probable is selected with argmax.

possible actions to move or not to move a 3D object depending on the six degrees of freedom for rigid displacement in space. The 13 possible actions divide into six pose actions for positive parameter adjustment, six for negative changes and an action to stop the process (i.e. not to move the object). For each of these actions, we set units depending on an image and a current crop: While movements for $t_x$, $t_y$ are measured in pixels and determine movements of the bounding box as such, $r_x$, $r_y$, $r_z$ are measured in degrees and $t_z$ is determined as the diameter in pixels of the current bounding box. While an action can change the position and size of the crop, the image crop is always rescaled to a quadratic $n \times n$ patch of the same size as the rendering.

We decide to implement the action decision CNN with a lightweight architecture that allows for training on a consumer laptop (see Section 4). The architecture details are shown in Fig. 2. An attention mechanism is implemented as guidance for the network to focus on relevant image regions and ignore occlusions. This attention map is learnt in an unsupervised way during training to mask the embedded feature tensor and realize a weighted global average pooling. We train the model end to end with synthetic data (details in Sect. 4) where a random action vector is created, normalized and a softmax cross entropy loss between logits and labels is utilized to optimize for the probability error in this mutually exclusive and discrete action classification task.

**Stopping Criteria & Tracking Mode.** Usually the iteration process is stopped with the stop action in frame $t$ and the last pose is used to initialize the process in frame $t+1$. As we discretize the pose steps, the stop criterion, however, may not always be hit perfectly. Moreover, the decision boundary between the stop criterion and some close action may not always be clear in every case leading to oscillations between two or multiple predictions close to the correct result. To cope with this in practice, we can also stop the process early if we encounter oscillations and if an intermediate pose has been predicted before in the same loop or if a maximum number of iterations is reached.


Fig. 3: **Dataset Creation.** High quality 3D models from YCB [81] and 3D models from Linemod [25] are rendered in various poses on top of 2D images from MS COCO [46]. Augmentation in form of blur, light changes and occlusions are added. A comparison image from the real dataset is shown on the right.

## 4 Experiments

We implemented the model using the 3D renderer from unity [75] with a customized version of the ML-agent toolkit [32] to seamlessly support our model, load training, provide visualizations for debugging purposes and run all our experiments. We combined it with TensorFlow (1.7.1 tensorflow-gpu) and used TensorFlow 1.10.0 for training to have the necessary functionality support. The batch size is set to 32 and we use the ADAM [39] optimizer with a learning rate of $10^{-4}$ and exponential decay of 5% every 1k iterations. We trained all our models until convergence (i.e. 25k iterations for object-specific training and 50k for multi-object training). For all our experiments as well as training and dataset creation, we used a consumer laptop with an Intel Xeon E3-1505Mv6 CPU and an Nvdia Quadro P5000 mobile GPU.

### 4.1 Training on Synthetic Data

To train our model, we create a synthetic dataset generation pipeline where we render the 3D models with changing backgrounds and varying poses in clutter and occlusion on top of real images. Following [35] we use images from the MS COCO [46] dataset as background. We randomly pick 40k images from [46] and use the high quality 3D models from YCB [81] and the models from Linemod [25] to render the objects during training in various poses on top of the images as shown in Fig. 3.

**Data Augmentation.** We augment the renderings in different ways with occluders, crops, image blur as well as material and light changes before placing it on top of the COCO images. As our network operates on cropped images patches of size $128 \times 128$ pixels, we perform the augmentation on these patches, too. Some augmentation results are shown in Fig. 3. We synthetically generate 50k images for each YCB [81] object and 50k images for each Linemod [25] model. The augmentation pipeline is described in detail in the supplementary material. We consider these images as our synthetic ground truth.
To simulate also the initial pose seeds, we produce a variety of 3D renderings

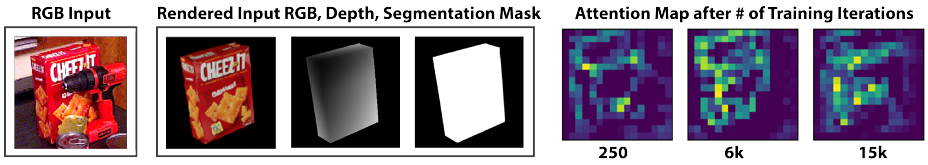


Fig. 4: **Unsupervised Training of Attention Map.** The input RGB as well as the three input renderings are shown together with the results of the unsupervised training of the attention map after different numbers of training steps.

without any augmentation a set of actions away from the related synthetic ground truth patch. We want our method to work particularly well close to the correct result where it is crucial to take the right decisions in order to converge. For this reason instead of rendering random seeds evenly distributed in pose space, we pay close attention near the ground truth by providing more training data in this region. We group the pose seeds in five clusters: 10k each for YCB and Linemod. The first cluster contains *small* misalignment in only one action direction, where each action has an equal chance of $1/13$ to be picked, also the stop-action. For the step size it holds $t_x, t_y \in [1, 5]$, $t_z, r_i \in [1, 4] \, \forall i$. The second group consists of *larger* misalignment in only one direction with equal chance. For this we chose $t_x, t_y \in [5, 30]$, $t_z \in [1, 15]$, $r_i \in [4, 20] \, \forall i$. The third group is *mixed* where we have one larger misalignment in one direction and the remaining actions are random small misalignment (e.g. $t_x = 10$ and all other directions are randomly chosen as in group one). The fourth and fifth groups are a *random small* and a *random large* mix of misalignments from groups one and two.

**Training.** We train networks for each YCB [81] model (object-specific training) and one network with mixed training including all YCB and Linemod models (multi-object training). Fig. 4 shows the unsupervised training of our attention map on the same image after different number of iterations for training with cracker box. It can be seen, that after attention on high gradient object regions (250 iterations), the mask emphasizes on the overall object geometry excluding big occlusion patches (6k iterations) before it learns to exclude the finer occluder details such as the front part of the drill (15k iterations).

### 4.2 Pose Estimation & Dataset Quality

**Datasets.** High quality pose annotations are usually acquired with fiducial markers, manual annotation or a combination of both [25,5]. This process is very time-consuming and thus video annotations for 6D pose estimation is not easily retrieved. In order to produce the marker-free video pose dataset YCB [81], the authors manually annotated only the poses of all the objects in the first frame of

a sequence and refine them with an algorithm based on Signed Distance Functions. The ground truth labels for the rest of the frames within the sequences are retrieved by camera trajectory estimation with a depth-based tracker and the constraint for constant relative object poses within the scene. This eliminates possible fiducial marker cues that could eventually provide a signal to a learning-based method at the cost of not being able to freely move the objects. While this allows also for larger frame sets, the quality of the annotations can vary. The Laval [22] video dataset circumvents this issue through the use of a motion capture system and retroreflective markers attached to the real objects in the scene. A post-processing step in their pipeline cures the depth images by removing strong artifacts that arise from marker reflections to provide cleaned depth images also for RGBD methods. We test our models on these two datasets and evaluate both quantitatively and qualitatively. We note that the models in the YCB dataset are part of our training, while the objects from Laval are entirely *unseen*.

**Quantitative & Qualitative Evaluation.** For all quantitative experiments, we follow the protocol of [21,22] and reset the pose estimation with the annotated pose every 15 frames. The maximum number of action steps per frame is set to 30. At first, we test our networks trained on individual YCB models and compare with their ground truth poses [81]. The result is reported in comparison with the state-of-the-art [81,20,30,53] in Tab. 1 column two to six. We utilize the 3D metrics for ADD and ADI (for symmetric objects) relative to the object diameter as proposed in [25]. A further comparison with absolute thresholds is provided in the supplementary material.
We can note an average improvement of 9.94% compared to [53] for our method and investigated the failure cases. While most of them seemed visually plausible, we still observed a significant accuracy variance between the video sequences in YCB which we further analyzed. It turned out that the annotations for some of the objects are slightly shifted as shown in Fig. 5. Our method – in contrast to others with which we compare in Tab. 1 – is fully trained on synthetic data. Thus, we cannot learn an annotation offset during training time due to the fact that our training setup provides pixel-perfect ground truth. Further investigations revealed that the ground truth annotation quality is a common issue amongst multiple videos sequences in this dataset.
We believe that the main source for this is that an incorrect annotation in the first frame propagates constantly through the whole sequence, and the manual label was only given in frame one [81]. We correct this shift such that the annotation visually overlaps the RGB observation by one single, constant translation delta for each of the sequences and rerun the evaluation. The results are shown in the last column of Tab. 1, where also the accuracy of our method improves significantly to a margin of 28.64% over the state-of-the-art. The corrected annotations will be made publicly available to ease the comparison between synthetic and real data training on this dataset also for others and to help improving fu-

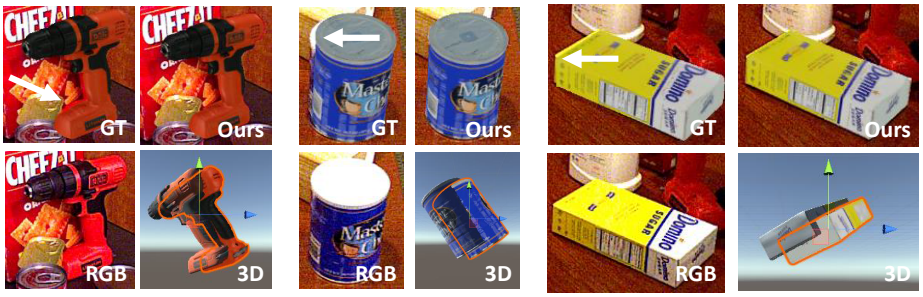


Fig. 5: **Annotation Quality for YCB.** The input image is shown together with our prediction and the ground truth annotations. Arrows and 3D visualization are added to detail the difference in these cases where our estimation is considered incorrect.

ture pipelines.

**Generalization and Ablation.** Given these problematic initial annotations, we refrain form further interpretation of the results and investigate another dataset [22]. To the best of our knowledge, we are the first RGB-only method to report object-specific results on the challenging sequences of Laval [22] where we test the generalization capabilities of our multi-object model. Please note that the objects of the dataset have not been seen during training. The results are summarized in Tab. 2 where we also ablate the rendered depth input channel, and Fig. 7 shows an example scenario. We follow the evaluation protocol of [22] and report separately the average error for translation and rotation. Tab. 2 shows that our multi-object model generalizes well on this dataset where the ground truth is acquired with a professional tracking system. Both models are able to track the unseen object in translation. While the full model provides close results both for translation and rotation, the ablated model focuses only on the translation component and predicts stop once the object centre is aligned with only weak corrections for the rotation. Without the depth rendering, the rotational error is significantly larger. Rendering the synthetic depth helps with respect to the rotational accuracy. This can be explained by the fact that moving the object in a close proximity to the observation does not require detailed understanding of depth while rotating it correctly is more intricate. In practice we observed that the network first aligns the object in $t_x$ and $t_y$ before correcting rotations and $t_z$ values. We leverage this observation for initialization in Section 4.4.

### 4.3 Robustness & Convergence

The performance of conventional trackers largely depends on the difference between the correct pose and the initialization [1]. As their paradigm is temporally consistent motion in the videos, oftentimes close-to-correct poses are available from the result of the previous frame or re-initialize with another algorithm [13].

| Model | PC [81] | HMP [20] | SD [30] | HM [53] | Ours OS | Ours + Shift |
|---|---|---|---|---|---|---|
| 002_master_chef_can | 3.60 | 40.10 | 33.00 | **75.80** | 7.70 | 91.88 |
| 003_cracker_box | 25.10 | 69.50 | 46.60 | 86.20 | **88.36** | 97.76 |
| 004_sugar_box | 40.30 | 49.70 | **75.60** | 67.70 | 58.35 | 91.95 |
| 005_tomato_soup_can | 25.50 | 36.10 | **40.80** | 38.10 | 38.23 | 57.99 |
| 006_mustard_bottle | 61.90 | 57.90 | 70.60 | **95.20** | 87.74 | 98.49 |
| 007_tuna_fish_can | 11.40 | 9.80 | 18.10 | 5.83 | **47.90** | 52.89 |
| 008_pudding_box | 14.50 | 67.20 | 12.20 | **82.20** | 58.68 | 76.00 |
| 009_gelatin_box | 12.10 | 59.10 | 59.40 | **87.80** | 37.08 | 89.20 |
| 010_potted_meat_can | 18.90 | 42.00 | 33.30 | **46.50** | 45.99 | 60.61 |
| 011_banana | 30.30 | 19.30 | 16.60 | 30.80 | **74.02** | 90.43 |
| 019_pitcher_base | 15.60 | 58.50 | 90.00 | 57.90 | **99.40** | 100.00 |
| 021_bleach_cleanser | 21.20 | 69.40 | 70.90 | 73.30 | **95.04** | 95.30 |
| *024_bowl* | 12.10 | 27.70 | 30.50 | 36.90 | **99.44** | 99.44 |
| 025_mug | 5.20 | 12.90 | 40.70 | 17.50 | **45.35** | 76.59 |
| 035_power_drill | 29.90 | 51.80 | 63.50 | **78.80** | 52.77 | 97.35 |
| *036_wood_block* | 10.70 | 35.70 | 27.70 | 33.90 | **52.28** | 63.48 |
| 037_scissors | 2.20 | 2.10 | 17.10 | 43.10 | **63.33** | 81.11 |
| 040_large_marker | 3.40 | 3.60 | 4.80 | 8.88 | **39.53** | 41.73 |
| *051_large_clamp* | 28.50 | 11.20 | 25.60 | 50.10 | **64.01** | 82.83 |
| *052_extra_large_clamp* | 19.60 | 30.90 | 8.80 | 32.50 | **88.02** | 91.37 |
| *061_foam_brick* | 54.50 | 55.40 | 34.70 | 66.30 | **80.83** | 80.83 |
| Average | 21.26 | 38.57 | 39.07 | 53.11 | **63.05** | 81.75 |

Table 1: Evaluation on the YCB dataset with our object-specific models. We compare the percentage of frames for which the 3D AD{D|I} error is $< 10\%$ of the object diameter [25]. Symmetric objects are shown in italic letters.

Their sensitivity to the initialization results in drift or tracking loss if the seed pose is too far off. Recent methods severely suffer, for instance, if the bounding box overlap is below 50% [21,22]. Moreover, most conventional 3D trackers are not able to detect whether their estimation is correct or not. In contrast to these methods, we propose a pose estimation pipeline with a large convergence basin that is able to detect its own drift by analysing the number of steps and our stopping criterion.

We test the convergence radius of our model by providing different initial poses with gradually increasing deviation from the correct result. After manually checking the ground truth poses of the YCB dataset [81], we decided to test with power drill on all keyframes from video sequence 50 which provides reliable annotations. We prepare initial poses by deteriorating the ground truth annotations with increasing noise from the correct result to an initialization which is 270 actions apart. This is done by adding actions to the GT pose with the state $[t_x, t_y, t_z, r_x, r_y, r_z]$ in the form of:

$$\Delta \cdot [m(t_x), m(t_y), m(t_z), m(r_x), m(r_y), m(r_z)]\,, \tag{1}$$

| | Ours full | | | | Ours w/o D | | | |
|---|---|---|---|---|---|---|---|---|
| **Occlusion** | 0% | 15% | 30% | 45% | 0% | 15% | 30% | 45% |
| **Turtle** | | | | | | | | |
| T[mm] | 5.92 | 9.91 | 12.91 | 23.92 | 5.53 | 6.37 | 16.14 | 12.63 |
| R[deg] | 7.09 | 14.87 | 14.87 | 14.11 | 18.31 | 20.13 | 26.03 | 24.97 |
| **Walkman** | | | | | | | | |
| T[mm] | 8.74 | 18.93 | 31.98 | 45.13 | 11.63 | 15.63 | 20.12 | 31.30 |
| R[deg] | 6.97 | 11.33 | 21.17 | 22.26 | 40.68 | 44.47 | 50.18 | 45.14 |

Table 2: Evaluation result on Laval dataset for different levels of noise. We compare the full model to a model without rendered depth input. More objects are investigated in the supplementary material.

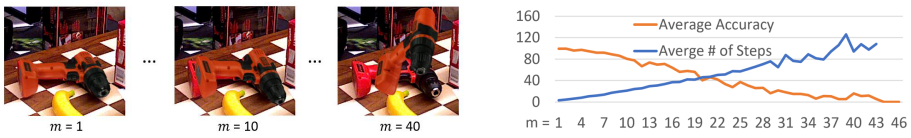


Fig. 6: **Robustness Test.** The average ADD score is shown for increasing deviations (examples on left) from the ground truth (orange) while the average number of steps the method needed for convergence is illustrated in blue. For deviations with $m \geq 43$ the method did not converge within 200 steps.

$$\text{where} \quad m(s) = m \cdot \operatorname{sgn}(X), \tag{2}$$

for all state variables $s$. We vary the value $m \in \{0, ..., 45\}$ and $X$ is drawn from the uniform distribution $U(-1, 1)$ and determines the direction of corruption. The parameter $\Delta = 6$ sets the stepsize for our test.

We use the individually trained model and set the stepsize for all actions to three. Then we run the method and record the average ADD accuracy score as well as the average number of steps in case the model converges to the correct solution. We randomly reduce the amount of keyframes for $m \in \{25, ..., 30\}$ to 25% and for $m \in \{31, ..., 45\}$ to 10% to avoid unreasonably long computations. If convergence is not reached within 200 steps, we treat the run as a fail. The results are summarized in Fig. 6. Note that even for a large deviation of $m = 12$ which is significantly larger then the deviation found in the video sequence, our accuracy is ADD = 73.8%. Moreover, we can also see reasonable convergence in cases with 50% or fewer bounding box overlap where other methods [21] struggle and drift.

We use this wide convergence basin to show that our framework can be modified without retraining to also provide an initial pose close to the correct one (cf. Sec. 4.4).

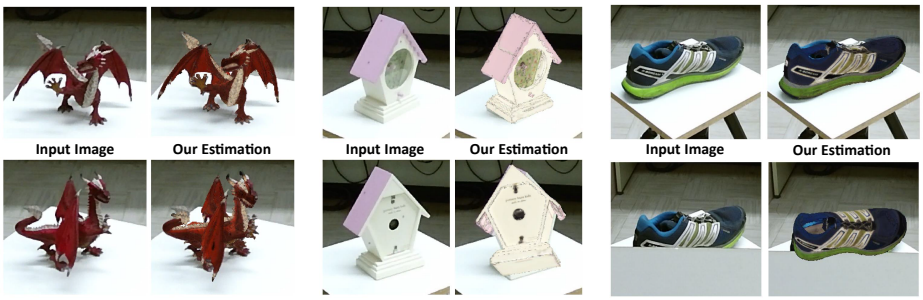


Fig. 7: **Estimation Examples.** We show some prediction examples from the Laval [22] dataset. Self-occluded fine details (left), low texture (middle) and occlusions (right) can cause pose estimation failure for unseen objects.

**Failure Cases.** Even though the convergence of our method is reliable in most cases, the network capacity is limited. This results in pose estimation failures in case of heavy occlusions and fine detailed geometry. Moreover, we share the issue with other RGB-only methods that low-textured objects are difficult to estimate reliably which results in drift in some cases as depicted in Fig. 7 together with further examples.

**Runtime.** The structure of our approach allows for automatic dynamic runtime improvements depending on the motion present in the scene. Since the number of iteration steps is non-static and the 3D rendering is negligible for this comparison, the overall runtime depends on two parameters: the action decision cycle and the number of actions. In our current implementation, the runtime for one loop in the cycle breaks down in the image preprocessing done on CPU and the inference on the GPU. We performed a runtime test averaging 512 iterations. The results are shown in Tab. 3.

| Average Runtime on | CPU | GPU | Total |
|---|---|---|---|
| Average Runtime in ms | 14.6 | 5.2 | 19.8 |

Table 3: Average Runtime of Action Decision Process Cycle.

Given the average of 4.2 actions on our YCB tests, we report an overall average runtime of 83.16 ms or 12 FPS. Note that the runtime could be increased if the image processing was also ported to the GPU.

### 4.4 Initialization & Detection

Tracking is often done by detection [12,81] or with the help of a depth map [22]. However, Deng et al. [13] recently proposed a RGB-only tracking solution.

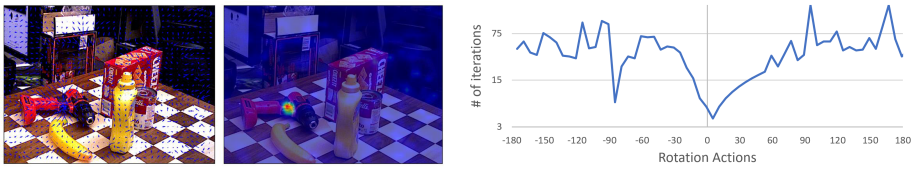


Fig. 8: **Initial Point & Rotation Seeding.** The predictions for $t_x$ and $t_y$ generate a vector field over the image (left) whose divergence (middle) determines the initial point. Seeding a random rotation at this point allows to calculate the initial pose. The necessary number of iterations is plotted (right) against different seeds at a certain deviation from this rotation in just one action parameter (in this case $r_z$). A good initialization in the example is +5 actions away where the curve has its minimum.

Other pose refinement models like [49] for [35] or [43] for [81] require an initial detector. We empirically observed that the model tends to first align the rendering for the translation and performs rotation actions afterwards. We make use of this observation and run the network with multiple seeds as a pose detection pipeline omitting the use of another model. For this, we randomly chose an object pose and seed the image at different locations by changing $t_x$ and $t_y$ for the pose. We then run one iteration of the network in every location and record just the values for $t_x$ and $t_y$. We normalize the 2-vector given these inputs and generate a sparse vector field $\mathbf{V}$ on top of the image as shown in Fig. 8 where we place these vectors at the seed centres. This vector field is rather random for non-overlapping regions while its flux points toward the projection centre of the object if visible. Applying a divergence operation $\mathrm{W} = \nabla \cdot \mathbf{V}$ on the smoothed vectors allows to find the object centre as the maximum of W while analyzing W helps also to determine a valuable bounding box size for a first crop. Running the method on a coarsely discretized rotation space in this crop allows to find an initial rotation as shown in Fig. 8 where the minimum number of iteration positively correlates with a possible starting rotation. As the initial seeds can be calculated independent from each other, this process can be parallelized.

## 5 Conclusion

We reformulated 6D pose estimation as an action decision process and presented a pipeline to solve it as a generic task without the need for object-specific training. The method implements a dynamic runtime complexity depending on the inter-frame motion to increase runtime performance and generalizes to unseen objects. However, while improving the state-of-the-art for RGB-based video pose estimation, it still struggles in challenging cases for unseen objects. Currently we search for the next best pose in every step. An interesting direction for future research could be to integrate built up knowledge over time leveraging e.g. reinforcement learning.

## References


1. Akkaladevi, S., Ankerl, M., Heindl, C., Pichler, A.: Tracking multiple rigid symmetric and non-symmetric objects in real-time using depth data. In: 2016 IEEE International Conference on Robotics and Automation (ICRA). pp. 5644–5649. IEEE (2016)
2. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: European conference on computer vision. pp. 404–417. Springer (2006)
3. Besl, P.J., McKay, N.D.: Method for registration of 3-d shapes. In: Sensor fusion IV: control paradigms and data structures. vol. 1611, pp. 586–606. International Society for Optics and Photonics (1992)
4. Birdal, T., Dobryden, I., Ilic, S.: X-tag: A fiducial tag for flexible and accurate bundle adjustment. In: 3D Vision (3DV), 2016 Fourth International Conference on. pp. 556–564. IEEE (2016)
5. Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6d object pose estimation using 3d object coordinates. In: European conference on computer vision. pp. 536–551. Springer (2014)
6. Busam, B., Esposito, M., Che'Rose, S., Navab, N., Frisch, B.: A stereo vision approach for cooperative robotic movement therapy. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 127–135 (2015)
7. Busam, B., Esposito, M., Frisch, B., Navab, N.: Quaternionic Upsampling: Hyperspherical Techniques for 6 DoF Pose Tracking. In: 2016 Fourth International Conference on 3D Vision (3DV). pp. 629–638. IEEE (10 2016). https://doi.org/10.1109/3DV.2016.71, `http://ieeexplore.ieee.org/document/7785139/`
8. Busam, B., Hog, M., McDonagh, S., Slabaugh, G.: Sterefo: Efficient image refocusing with stereo vision. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 0–0 (2019)
9. Busam, B., Ruhkamp, P., Virga, S., Lentes, B., Rackerseder, J., Navab, N., Hennersperger, C.: Markerless inside-out tracking for 3d ultrasound compounding. In: Simulation, Image Processing, and Ultrasound Systems for Assisted Diagnosis and Navigation, pp. 56–64. Springer (2018)
10. Cai, H., Werner, T., Matas, J.: Fast detection of multiple textureless 3-d objects. In: International Conference on Computer Vision Systems. pp. 103–112. Springer (2013)
11. Capellen, C., Schwarz, M., Behnke, S.: Convposecnn: Dense convolutional 6d object pose estimation. arXiv preprint arXiv:1912.07333 (2019)
12. Crivellaro, A., Rad, M., Verdie, Y., Moo Yi, K., Fua, P., Lepetit, V.: A novel representation of parts for accurate 3d object detection and tracking in monocular images. In: Proceedings of the IEEE international conference on computer vision. pp. 4391–4399 (2015)
13. Deng, X., Mousavian, A., Xiang, Y., Xia, F., Bretl, T., Fox, D.: Poserbpf: A rao-blackwellized particle filter for 6d object pose tracking. arXiv preprint arXiv:1905.09304 (2019)
14. Do, T.T., Cai, M., Pham, T., Reid, I.: Deep-6DPose: Recovering 6D Object Pose from a Single RGB Image. arXiv preprint arXiv:1802.10367 (2018)
15. Doumanoglou, A., Kouskouridas, R., Malassiotis, S., Kim, T.K.: Recovering 6d object pose and predicting next-best-view in the crowd. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3583–3592 (2016)

16. Drost, B., Ulrich, M., Bergmann, P., Hartinger, P., Steger, C.: Introducing mvtec itodd-a dataset for 3d object recognition in industry. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2200–2208 (2017)
17. Esposito, M., Busam, B., Hennersperger, C., Rackerseder, J., Lu, A., Navab, N., Frisch, B.: Cooperative robotic gamma imaging: Enhancing us-guided needle biopsy. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 611–618. Springer (2015)
18. Esposito, M., Busam, B., Hennersperger, C., Rackerseder, J., Navab, N., Frisch, B.: Multimodal US–gamma imaging using collaborative robotics for cancer staging biopsies. International journal of computer assisted radiology and surgery **11**(9), 1561–1571 (2016)
19. Fiala, M.: ARTag, a fiducial marker system using digital techniques. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. vol. 2, pp. 590–596. IEEE (2005)
20. Fu, M., Zhou, W.: Deephmap++: Combined projection grouping and correspondence learning for full dof pose estimation. Sensors **19**(5), 1032 (2019)
21. Garon, M., Lalonde, J.F.: Deep 6-dof tracking. IEEE transactions on visualization and computer graphics **23**(11), 2410–2418 (2017)
22. Garon, M., Laurendeau, D., Lalonde, J.F.: A framework for evaluating 6-dof object trackers. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 582–597 (2018)
23. Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J., Mar\'\in-Jiménez, M.J.: Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recognition **47**(6), 2280–2292 (2014)
24. Held, R., Gupta, A., Curless, B., Agrawala, M.: 3d puppetry: a kinect-based interface for 3d animation. In: UIST. pp. 423–434. Citeseer (2012)
25. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: Asian conference on computer vision. pp. 548–562. Springer (2012)
26. Hodan, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., Zabulis, X.: T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 880–888. IEEE (2017)
27. Hodan, T., Michel, F., Brachmann, E., Kehl, W., GlentBuch, A., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., et al.: Bop: Benchmark for 6d object pose estimation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 19–34 (2018)
28. Hodaň, T., Zabulis, X., Lourakis, M., Obdržálek, Š., Matas, J.: Detection and fine 3d pose estimation of texture-less objects in rgb-d images. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4421–4428. IEEE (2015)
29. Holynski, A., Kopf, J.: Fast depth densification for occlusion-aware augmented reality. In: SIGGRAPH Asia 2018 Technical Papers. p. 194. ACM (2018)
30. Hu, Y., Hugonot, J., Fua, P., Salzmann, M.: Segmentation-driven 6d object pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3385–3394 (2019)
31. Joseph Tan, D., Tombari, F., Ilic, S., Navab, N.: A versatile learning-based 3d temporal tracker: Scalable, robust, online. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 693–701 (2015)

32. Juliani, A., Berges, V.P., Vckay, E., Gao, Y., Henry, H., Mattar, M., Lange, D.: Unity: A general platform for intelligent agents. arXiv preprint arXiv:1809.02627 (2018)
33. Kaskman, R., Zakharov, S., Shugurov, I., Ilic, S.: Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. In: The IEEE International Conference on Computer Vision (ICCV) Workshops (Oct 2019)
34. Kato, H., Billinghurst, M.: Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In: Augmented Reality, 1999.(IWAR'99) Proceedings. 2nd IEEE and ACM International Workshop on. pp. 85–94. IEEE (1999)
35. Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In: Proceedings of the International Conference on Computer Vision (ICCV 2017), Venice, Italy. pp. 22–29 (2017)
36. Kehl, W., Milletari, F., Tombari, F., Ilic, S., Navab, N.: Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation. In: European Conference on Computer Vision. pp. 205–220. Springer (2016)
37. Kehl, W., Tombari, F., Ilic, S., Navab, N.: Real-time 3d model tracking in color and depth on a single cpu core. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 745–753 (2017)
38. Kehl, W., Tombari, F., Navab, N., Ilic, S., Lepetit, V.: Hashmod: A hashing method for scalable 3d object detection. In: BMVC. vol. 1, p. 2 (2015)
39. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)
40. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics **36**(4) (2017)
41. Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: An Accurate O(n) Solution to the PnP Problem. International Journal of Computer Vision **81**(2), 155–166 (2 2009). https://doi.org/10.1007/s11263-008-0152-6, `http://link.springer.com/10.1007/s11263-008-0152-6`
42. Leutenegger, S., Chli, M., Siegwart, R.Y.: BRISK: Binary robust invariant scalable keypoints. In: Computer Vision (ICCV), 2011 IEEE International Conference on. pp. 2548–2555. IEEE (2011)
43. Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: Deepim: Deep iterative matching for 6d pose estimation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 683–698 (2018)
44. Li, Y., Snavely, N., Huttenlocher, D.P.: Location recognition using prioritized feature matching. In: European conference on computer vision. pp. 791–804. Springer (2010)
45. Li, Z., Wang, G., Ji, X.: Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
46. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
47. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2), 91–110 (2004)
48. Manhardt, F., Arroyo, D.M., Rupprecht, C., Busam, B., Birdal, T., Navab, N., Tombari, F.: Explaining the ambiguity of object detection and 6d pose from visual data. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 6841–6850 (2019)

49. Manhardt, F., Kehl, W., Navab, N., Tombari, F.: Deep model-based 6d pose refinement in rgb. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 800–815 (2018)
50. Mian, A., Bennamoun, M., Owens, R.: On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. International Journal of Computer Vision **89**(2-3), 348–361 (2010)
51. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. IEEE transactions on robotics **31**(5), 1147–1163 (2015)
52. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Transactions on Robotics **33**(5), 1255–1262 (2017)
53. Oberweger, M., Rad, M., Lepetit, V.: Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 119–134 (2018)
54. Olson, E.: AprilTag: A robust and flexible visual fiducial system. In: Robotics and Automation (ICRA), 2011 IEEE International Conference on. pp. 3400–3407. IEEE (2011)
55. Pavlakos, G., Zhou, X., Chan, A., Derpanis, K.G., Daniilidis, K.: 6-dof object pose from semantic keypoints. In: Robotics and Automation (ICRA), 2017 IEEE International Conference on. pp. 2011–2018. IEEE (2017)
56. Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H.: Pvnet: Pixel-wise voting network for 6dof pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4561–4570 (2019)
57. Periyasamy, A.S., Schwarz, M., Behnke, S.: Refining 6d object pose predictions using abstract render-and-compare. arXiv preprint arXiv:1910.03412 (2019)
58. Pitteri, G., Ilic, S., Lepetit, V.: Cornet: Generic 3d corners for 6d pose estimation of new objects without retraining. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 0–0 (2019)
59. Rad, M., Lepetit, V.: BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In: ICCV (2017)
60. Rad, M., Oberweger, M., Lepetit, V.: Domain transfer for 3d pose estimation from color images without manual annotations. In: Asian Conference on Computer Vision. pp. 69–84. Springer (2018)
61. Rad, M., Oberweger, M., Lepetit, V.: Feature mapping for learning fast and accurate 3d pose inference from synthetic images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4663–4672 (2018)
62. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
63. Rennie, C., Shome, R., Bekris, K.E., De Souza, A.F.: A dataset for improved rgbd-based object detection and pose estimation for warehouse pick-and-place. IEEE Robotics and Automation Letters **1**(2), 1179–1185 (2016)
64. Richter-Klug, J., Frese, U.: Towards meaningful uncertainty information for cnn based 6d pose estimates. In: International Conference on Computer Vision Systems. pp. 408–422. Springer (2019)
65. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: An efficient alternative to SIFT or SURF. In: 2011 International Conference on Computer Vision. pp. 2564–2571. IEEE (11 2011). https://doi.org/10.1109/ICCV.2011.6126544, `http://ieeexplore.ieee.org/document/6126544/`

66. Rusinkiewicz, S., Levoy, M.: Efficient variants of the icp algorithm. In: 3dim. vol. 1, pp. 145–152 (2001)
67. Rusu, R.B., Blodow, N., Marton, Z.C., Beetz, M.: Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments. In: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on. pp. 1–6. IEEE (2009)
68. Segal, A., Haehnel, D., Thrun, S.: Generalized-icp. In: Robotics: science and systems. vol. 2, p. 435. Seattle, WA (2009)
69. Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R.: Implicit 3D orientation learning for 6D object detection from RGB images. In: European Conference on Computer Vision. pp. 712–729. Springer (2018)
70. Tan, D.J., Ilic, S.: Multi-forest tracker: A chameleon in tracking. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2014)
71. Tejani, A., Tang, D., Kouskouridas, R., Kim, T.K.: Latent-class hough forests for 3D object detection and pose estimation. In: European Conference on Computer Vision. pp. 462–477. Springer (2014)
72. Tekin, B., Sinha, S.N., Fua, P.: Real-time seamless single shot 6D object pose prediction. CVPR (2018)
73. Tombari, F., Salti, S., Di Stefano, L.: Unique signatures of histograms for local surface description. In: European conference on computer vision. pp. 356–369. Springer (2010)
74. Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., Birchfield, S.: Deep object pose estimation for semantic robotic grasping of household objects. arXiv preprint arXiv:1809.10790 (2018)
75. Unity Technologies: Unity. `https://unity3d.com/unity/whats-new/2018.3.6`, accessed: 2019-06-10
76. Wang, C., Xu, D., Zhu, Y., Martín-Martín, R., Lu, C., Fei-Fei, L., Savarese, S.: Densefusion: 6d object pose estimation by iterative dense fusion. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3343–3352 (2019)
77. Wang, T., Ling, H.: Gracker: A graph-based planar object tracker. IEEE transactions on pattern analysis and machine intelligence **40**(6), 1494–1501 (2017)
78. Wang, Y., Jin, S., Ou, Y.: A multi-task learning convolutional neural network for object pose estimation. In: International Conference on Robotics and Biomimetics (ROBIO). pp. 284–289. IEEE (2019)
79. Wohlhart, P., Lepetit, V.: Learning descriptors for object recognition and 3d pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3109–3118 (2015)
80. Wu, C., Fraundorfer, F., Frahm, J.M., Pollefeys, M.: 3D model search and pose estimation from single images using VIP features. In: Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on. pp. 1–8. IEEE (2008)
81. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. Robotics: Science and Systems (RSS) (2018)
82. Yuheng Ren, C., Prisacariu, V., Murray, D., Reid, I.: Star3d: Simultaneous tracking and reconstruction of 3d objects using rgb-d data. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1561–1568 (2013)
83. Yun, S., Choi, J., Yoo, Y., Yun, K., Young Choi, J.: Action-decision networks for visual tracking with deep reinforcement learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)

84. Zakharov, S., Shugurov, I., Ilic, S.: Dpod: 6d pose object detector and refiner. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
85. Zhang, Z.: Iterative point matching for registration of free-form curves and surfaces. International journal of computer vision **13**(2), 119–152 (1994)
86. Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the continuity of rotation representations in neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5745–5753 (2019)

# I Like to Move It - Supplementary Material

Benjamin Busam, Hyun Jun Jung, and Nassir Navab

Technical University of Munich

## A Data Augmentation Details

We simulate two different blurs to augment the data with TensorFlow. In 75% of the cases, we randomly add motion blur and in 25% of training scenarios a radial blur. Both are generated with a mean of $\mu = 0$ and $\sigma = 0.05$ standard deviation for all three colour channels. Variety in the exposures are augmented through changes of brightness, contrast and saturation values in the range of $[0.95, 1.25]$. For object material and light augmentation, we leverage the unity engine and simulate 20% of unlit material and 80% of standard material (i.e. metallic with $[0, 0.85]$ and glossiness/smoothness with $[0, 0.8]$). Light is augmented with five point lights at random positions with an intensity drawn from $[0.5, 1.5]$. We change the light colour randomly by picking one colour from $C = \{\text{blue, cyan, green, magenta, red, yellow, white}\}$ at every capture and set the same colour for all the five lights. The colour brightness for the light is randomly enhanced offering subtle additional variation in contrast to the intensity changes. Then we randomly crop the rendering patch with $128 \times 128$ pixels to a height and width within $[96, 128]$ and resize the patch to a value within $[32, 64]$. To simulate occlusion, we render 20k patches from YCB and Linemod models with random poses from which we pick four samples at each training step. Firstly, they all are processed by the aforementioned blur and colour augmentation scheme. In 50% of the cases, we will not occlude the patch. In the other cases we will use these four samples for occlusion. With a 12.5% chance we respectively select either one, two or three occluders at random or use all four. Finally, we crop the entire masked region of the augmentation pipeline in 25% of the cases to simulate another occlusion scenario where we select the cropped region patch height and width randomly from $[72, 96]$. We apply this procedure to generate 50k images for each YCB [81] object and 50k images for each Linemod [25] model.

## B Additional YCB Comparison

The main paper shows a quantitative evaluation on the standard ADD metric [25] relative to the object diameter where a pose estimate is considered successful if its ADD value is below 10% of the object diameter. The final ADD score is calculated by the percentage of frames with such a successful estimation. Tables 1, 2, 3 additionally compare the area under the ADD threshold curve (AUC) for varying absolute thresholds from zero to 0.1 m [81]. The extensive study in comparison with the state-of-the-art shows that our method compares favourable on the standard benchmark (Ours OS) and significantly better with the shift-correction.

| Model | 3DC [81] | PC [81] | CPC [11] | PRBPF [13] | Ours OS | + Shift |
|---|---|---|---|---|---|---|
| 002_master_chef_can | 12.30 | 50.90 | 62.32 | 63.30 | 65.61 | 91.15 |
| 003_cracker_box | 16.80 | 51.70 | 66.69 | 77.80 | 84.34 | 90.74 |
| 004_sugar_box | 28.70 | 68.60 | 67.19 | 79.60 | 78.43 | 91.05 |
| 005_tomato_soup_can | 27.30 | 66.00 | 75.52 | 73.00 | 66.83 | 76.06 |
| 006_mustard_bottle | 25.90 | 79.90 | 83.79 | 84.70 | 86.05 | 94.03 |
| 007_tuna_fish_can | 5.40 | 70.40 | 60.98 | 64.20 | 65.90 | 69.12 |
| 008_pudding_box | 14.90 | 62.90 | 62.17 | 64.50 | 79.00 | 83.01 |
| 009_gelatin_box | 25.40 | 75.20 | 83.84 | 83.00 | 82.92 | 92.78 |
| 010_potted_meat_can | 18.70 | 59.60 | 65.86 | 51.80 | 75.21 | 79.44 |
| 011_banana | 3.20 | 72.30 | 37.74 | 18.40 | 84.99 | 90.19 |
| 019_pitcher_base | 27.30 | 52.50 | 62.19 | 63.70 | 85.14 | 94.22 |
| 021_bleach_cleanser | 25.20 | 50.50 | 55.14 | 60.50 | 89.27 | 90.68 |
| *024_bowl* | 2.70 | 6.50 | 3.55 | 28.40 | 85.89 | 87.03 |
| 025_mug | 9.00 | 57.70 | 45.83 | 77.90 | 78.95 | 87.83 |
| 035_power_drill | 18.00 | 55.10 | 76.47 | 71.80 | 76.56 | 91.95 |
| *036_wood_block* | 1.20 | 31.80 | 0.12 | 2.30 | 48.62 | 53.52 |
| 037_scissors | 1.00 | 35.80 | 56.42 | 38.70 | 79.78 | 83.99 |
| 040_large_marker | 0.20 | 58.00 | 55.26 | 67.10 | 73.27 | 75.31 |
| *051_large_clamp* | 6.90 | 25.00 | 29.73 | 38.30 | 56.09 | 65.97 |
| *052_extra_large_clamp* | 2.70 | 15.80 | 21.99 | 32.30 | 67.31 | 78.06 |
| *061_foam_brick* | 0.60 | 40.40 | 51.80 | 84.10 | 86.52 | 86.70 |
| Average | 13.02 | 51.74 | 53.55 | 58.35 | 76.03 | 83.47 |

Table 1: Evaluation on the YCB dataset with our object-specific models. We compare the area under the ADD threshold curve (AUC) for varying thresholds from zero to 0.1 m. Symmetric objects are shown in italic letters.

| Model | RKF [64] | HM [53] | R&C [57] | Dope [74] | Ours | OS + Shift |
|---|---|---|---|---|---|---|
| 002_master_chef_can | 54.60 | 81.90 | 76.70 | - | 65.61 | 91.15 |
| 003_cracker_box | 57.60 | 83.60 | 82.90 | 55.90 | 84.34 | 90.74 |
| 004_sugar_box | 84.10 | 82.10 | 86.40 | 75.70 | 78.43 | 91.05 |
| 005_tomato_soup_can | 68.30 | 79.80 | 57.40 | 76.10 | 66.83 | 76.06 |
| 006_mustard_bottle | 79.00 | 91.50 | 86.70 | 81.90 | 86.05 | 94.03 |
| 007_tuna_fish_can | 43.50 | 48.70 | 69.70 | - | 65.90 | 69.12 |
| 008_pudding_box | 50.30 | 90.20 | 68.80 | - | 79.00 | 83.01 |
| 009_gelatin_box | 74.80 | 93.70 | 73.00 | - | 82.92 | 92.78 |
| 010_potted_meat_can | 50.30 | 79.10 | 74.60 | 39.40 | 75.21 | 79.44 |
| 011_banana | 8.20 | 51.70 | 68.80 | - | 84.99 | 90.19 |
| 019_pitcher_base | 77.80 | 69.40 | 83.80 | - | 85.14 | 94.22 |
| 021_bleach_cleanser | 59.30 | 76.20 | 78.30 | - | 89.27 | 90.68 |
| *024_bowl* | - | 3.60 | 1.50 | - | 85.89 | 87.03 |
| 025_mug | 69.10 | 53.90 | 57.90 | - | 78.95 | 87.83 |
| 035_power_drill | 71.40 | 82.90 | 81.50 | - | 76.56 | 91.95 |
| *036_wood_block* | - | 0.00 | 0.00 | - | 48.62 | 53.52 |
| 037_scissors | - | 65.30 | 75.40 | - | 79.78 | 83.99 |
| 040_large_marker | - | 56.50 | 59.80 | - | 73.27 | 75.31 |
| *051_large_clamp* | - | 57.20 | 75.30 | - | 56.09 | 65.97 |
| *052_extra_large_clamp* | - | 23.60 | 20.40 | - | 67.31 | 78.06 |
| *061_foam_brick* | - | 32.10 | 37.00 | - | 86.52 | 86.70 |
| Average | 60.59 | 62.05 | 62.66 | 65.80 | 76.03 | 83.47 |

Table 2: Evaluation on the YCB dataset with our object-specific models. We compare the area under the ADD threshold curve (AUC) for varying thresholds from zero to 0.1 m. Symmetric objects are shown in italic letters.

| Model | HMP [20] | MT [78] | D-IM [43] | PV-N [56] | Ours OS | Ours OS + Shift |
|---|---|---|---|---|---|---|
| 002_master_chef_can | 75.80 | 62.70 | 71.20 | 81.60 | 65.61 | 91.15 |
| 003_cracker_box | 78.00 | 80.90 | 83.60 | 80.50 | 84.34 | 90.74 |
| 004_sugar_box | 76.50 | 83.80 | 94.10 | 84.90 | 78.43 | 91.05 |
| 005_tomato_soup_can | 72.10 | 60.40 | 86.10 | 78.20 | 66.83 | 76.06 |
| 006_mustard_bottle | 78.90 | 85.10 | 91.50 | 88.30 | 86.05 | 94.03 |
| 007_tuna_fish_can | 51.60 | 75.40 | 87.70 | 62.20 | 65.90 | 69.12 |
| 008_pudding_box | 85.60 | 17.70 | 82.70 | 85.20 | 79.00 | 83.01 |
| 009_gelatin_box | 86.70 | 79.90 | 91.90 | 88.70 | 82.92 | 92.78 |
| 010_potted_meat_can | 70.10 | 55.00 | 76.20 | 65.10 | 75.21 | 79.44 |
| 011_banana | 47.90 | 59.60 | 81.20 | 51.80 | 84.99 | 90.19 |
| 019_pitcher_base | 71.80 | 96.10 | 90.10 | 91.20 | 85.14 | 94.22 |
| 021_bleach_cleanser | 69.10 | 89.40 | 81.20 | 74.80 | 89.27 | 90.68 |
| *024_bowl* | - | 49.50 | 8.60 | - | 85.89 | 87.03 |
| 025_mug | 43.40 | 87.70 | 81.40 | 81.50 | 78.95 | 87.83 |
| 035_power_drill | 76.80 | 96.40 | 85.50 | 83.40 | 76.56 | 91.95 |
| *036_wood_block* | - | 43.80 | 60.00 | - | 48.62 | 53.52 |
| 037_scissors | 42.90 | 60.20 | 60.90 | 54.80 | 79.78 | 83.99 |
| 040_large_marker | 47.60 | 87.50 | 75.60 | 35.80 | 73.27 | 75.31 |
| *051_large_clamp* | - | 90.70 | 48.40 | - | 56.09 | 65.97 |
| *052_extra_large_clamp* | - | 88.10 | 31.00 | - | 67.31 | 78.06 |
| *061_foam_brick* | - | 26.30 | 35.90 | - | 86.52 | 86.70 |
| Average | 67.18 | 70.30 | 71.66 | 74.25 | 76.03 | 83.47 |

Table 3: Evaluation on the YCB dataset with our object-specific models. We compare the area under the ADD threshold curve (AUC) for varying thresholds from zero to 0.1 m. Symmetric objects are shown in italic letters.

## C Additional Laval Results

The main paper shows the results for the error on the Laval dataset [22] for two objects. Table 4 shows the results for the remaining objects of the dataset.

| | **Ours full** | | | | **Ours w/o D** | | | |
|---|---|---|---|---|---|---|---|---|
| **Occlusion** | 0% | 15% | 30% | 45% | 0% | 15% | 30% | 45% |
| **Clock** | | | | | | | | |
| T[mm] | 14.02 | 20.54 | 25.85 | 51.92 | 9.39 | 9.96 | 32.58 | 15.91 |
| R[deg] | 9.40 | 10.84 | 12.74 | 17.05 | 29.15 | 27.92 | 30.72 | 28.40 |
| **Cookie Jar** | | | | | | | | |
| T[mm] | 3.82 | 5.99 | 9.52 | 15.18 | 1.79 | 2.75 | 11.62 | 5.95 |
| R[deg] | 6.48 | 17.82 | 18.22 | 15.89 | 28.77 | 18.18 | 24.30 | 19.02 |
| **Dog** | | | | | | | | |
| T[mm] | 12.09 | 28.37 | 55.48 | 77.91 | 6.10 | 10.76 | 33.89 | 15.62 |
| R[deg] | 11.70 | 14.21 | 22.43 | 23.80 | 20.75 | 26.81 | 24.22 | 22.53 |
| **Dragon** | | | | | | | | |
| T[mm] | 22.47 | 29.39 | 36.37 | 40.06 | 25.69 | 25.13 | 27.71 | 30.65 |
| R[deg] | 3.34 | 4.89 | 11.65 | 13.39 | 27.16 | 36.40 | 37.61 | 30.94 |
| **Shoe** | | | | | | | | |
| T[mm] | 9.72 | 17.91 | 24.33 | 37.34 | 44.61 | 19.90 | 38.04 | 41.90 |
| R[deg] | 5.84 | 9.26 | 17.89 | 16.91 | 62.78 | 39.47 | 43.50 | 24.73 |
| **Watering Can** | | | | | | | | |
| T[mm] | 14.67 | 21.66 | 18.68 | 33.26 | 11.61 | 20.54 | 20.96 | 26.10 |
| R[deg] | 11.89 | 19.80 | 23.43 | 33.54 | 38.89 | 40.85 | 36.30 | 35.23 |

Table 4: Evaluation error on Laval dataset for different levels of noise. We compare the full model to a model without rendered depth input. While Turtle and Walkman are investigated in the main paper, this table shows the results for the remaining objects of the dataset.