How to track your dragon: A Multi-Attentional Framework for real-time RGB-D 6-DOF Object Pose Tracking

Isidoros Marougkas[1][0000−0003−2863−4155], Petros Koutras[1],[0000−0003−3183−2726], Nikos Kardaris[1], Georgios Retsinas[1][0000−0001−6734−3575], Georgia Chalvatzaki[2][0000−0002−5055−199X], and Petros Maragos[1][0000−0003−0534−2707]

[1]School of E.C.E., National Technical University of Athens, 15773, Athens, Greece
[2]Department of Computer Science TU Darmstadt, 64289, Darmstadt, Germany
Email: {ismarougkas, nick.kardaris}@gmail.com, {pkoutras, maragos}@cs.ntua.gr, gretsinas@central.ntua.gr, georgia@robot-learning.de

Abstract. We present a novel multi-attentional convolutional architecture to tackle the problem of real-time RGB-D 6D object pose tracking of single, known objects. Such a problem poses multiple challenges originating both from the objects' nature and their interaction with their environment, which previous approaches have failed to fully address. The proposed framework encapsulates methods for background clutter and occlusion handling by integrating multiple parallel soft spatial attention modules into a multitask Convolutional Neural Network (CNN) architecture. Moreover, we consider the special geometrical properties of both the object's 3D model and the pose space, and we use a more sophisticated approach for data augmentation during training. The provided experimental results confirm the effectiveness of the proposed multi-attentional architecture, as it improves the State-of-the-Art (SoA) tracking performance by an average score of 34.03 % for translation and 40.01 % for rotation, when tested on the most complete dataset designed, up to date, for the problem of RGB-D object tracking. Code will be available in: https://github.com/ismarou/How_to_track_your_Dragon

Keywords: Pose, Tracking, Attention, Geodesic, Multi-Task

1 Introduction

Robust, accurate and fast object pose estimation and tracking, i.e. estimation of the object's 3D position and orientation, has been a matter of intense research for many years. The applications of such an estimation problem can be found in Robotics, Autonomous Navigation, Augmented Reality, etc. Although the Computer Vision community has consistently studied the problem of object pose estimation and tracking for decades, the recent spread of affordable and reliable RGB-D sensors like Kinect, along with advances in Deep Learning (DL) and especially the use of CNNs as the new SoA image feature extractors, led to a new

Proc. European Conference on Computer Vision Workshops (ECCVW), August 2020

Sixth Int'l Workshop on Recovering 6D Object Pose (R6D2020), Glasgow, UK, August 23 2020

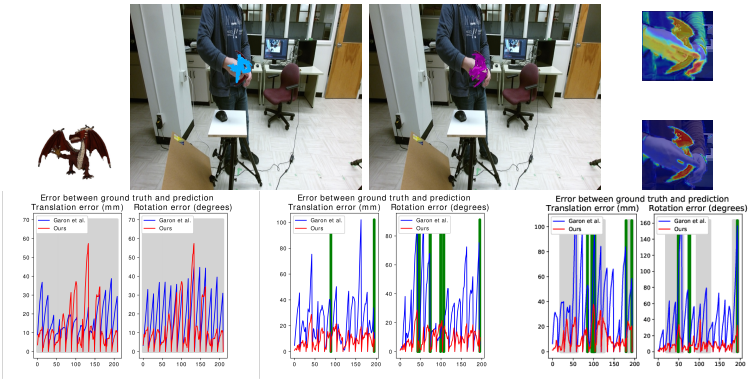

Fig. 1: **Top row:***(Left to right)* The "Dragon" model, an estimated pose in the "Hard Interaction" scenario for each of the SoA [8] *(light blue)* and our *(pink)* approaches and an example frame pair of Foreground extraction *(up)* and Occlusion handling *(down)* attention maps which are learned by minimizing the two auxiliary binary cross entropy losses. The following tradeoff occurs: as the occlusion increases, foreground attention, which focuses on the moving parts of the scene (i.e. the hand and the object), gets blurrier, while occlusion attention gets sharper and shifts focus from the object center to its body parts. **Bottom row:***(Left to right)* Translational and Rotational error plots of the SoA [8] *(blue)* and *(our)* approaches, for the "75% Vert. Occlusion", the "Rotation Only" and the "Hard Interaction" scenario, respectively. Grey regions stand for intervals of high occlusion and green ones for rapid movement.

era of research and a re-examination of several problems, with central aim the generalization over different tasks. CNNs have achieved ground-breaking results in 2D problems like object classification, object detection and segmentation. Thus, it has been tempting to the research community to increasingly use them in the more challenging 3D tasks, renouncing traditional algorithms.

The innate challenges of object pose estimation from RGB-D streams include background clutter, occlusions (both static, from other objects present in the scene, and dynamic, due to possible interactions with a human user), illumination variation, sensor noise, image blurring (due to fast movement) and appearance changes as the object viewpoint alters. Moreover, one should account for the pose ambiguity, which is a direct consequence of the object's own geometry, in possible symmetries, the challenges of proper parameter representation of rotations and the inevitable difficulties that an effort of forging a model faces, when extracting information about the 3D scene geometry from 2D-projected images.

In this paper, we build upon previous works [7,8], in order to face a series of those challenges that have not been fully resolved, so far. Thus, our main contributions are:

– An explicit background clutter and occlusion handling mechanism that leverages spatial attentions and provides an intuitive understanding of the tracker's region of interest at each frame, while boosting its performance. To the best of our knowledge, this is the first such strategy, that explicitly handles these two challenges, is incorporated into a CNN-based architecture, while achiev-

ing real-time performance. Supervision for this mechanism is extracted by fully exploiting the synthetic nature of our training data.
– The use of a novel multi-task pose tracking loss function, that respects the geometry of both the object's 3D model and the pose space and boosts the tracking performance by optimizing auxiliary tasks along with the principal one.
– SoA real-time performance in the hardest scenarios of the benchmark dataset [8], while achieving lower translation and rotation errors by an average of 34.03% for translation and 40.01% for rotation.

Accordingly, we provide the necessary methodological design details and experimental results that justify the importance of the proposed method in the challenging object pose tracking problem.

2 Related Work

Previous works attempt to tackle the problem using DL, focusing on two different directions: per-frame pose estimation (or, else, "tracking by detection") and temporal tracking.

Tracking-by-Detection The first family of proposed approaches in literature processes each video frame separately, without any feedback from the estimation of the previous timeframe. In [36], Xiang et al. constructed a CNN architecture that estimates binary object masks and then predicts the object class and its translation and rotation separately, while in [19], Kehl et al. extended the Single Shot Detection (SSD) framework [24] for 2D Object detection by performing discrete viewpoint classification for known objects. Finally, they refine their initial estimations via ICP [32] iterations. In [38], Zakharov et al. proposed a CNN framework that uses RGB images for pixel-wise object semantic segmentation in a mask-level. Following this, UV texture maps are estimated to extract dense correspondences between 2D images and 3D object models minimizing cross entropy losses. Those correspondences are used for pose estimation via P'n'P [22]. This estimation is, ultimately, inserted as a prior to a refinement CNN that outputs the final pose prediction. In PVNet [31], Peng et al. perform per-pixel voting-based regression to match 3D object coordinates with predefined keypoints inside the object surface, in order to handle occlusions. In [30], Pavlakos et al. extract semantic keypoints in single RGB images with a CNN and incorporate them into a deformable shape model. In [35], Wohlhart et al. employed a supervised contrastive convolutional framework to disentangle descriptors of different object instances and impose proportional distances to different poses of the same object. In [33], Sundermeyer et al. built a self-supervised Augmented Auto-Encoder that predicts 3D rotations only from synthetic data. In [29], Park et al. trained an adversiarially guided Encoder-Decoder to predict pixel-wise coordinates in a given image and then fed them to a P'n'P algorithm. More recently, iPose [18] is one of the attempts the philosophy of which is the closest to ours. Its authors segment binary masks with a pretrained MaskRCNN [11] to

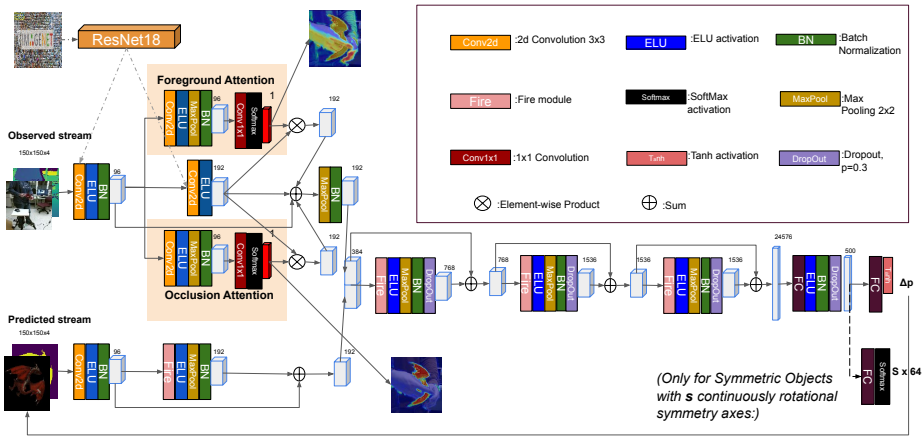

Fig. 2: Overview of the proposed CNN architecture for object pose tracking.

extract background clutter and occluders and they map 2D pixels to dense 3D object coordinates, which, in turn, are used as input to a P'n'P geometric optimization. Our attention modules have the same effect, but are computationally cheaper than MaskRCNN, as they relax the requirement for hard segmentation.

Temporal Tracking The second category under study is temporal tracking, where feedback is utilized, to allow for skipping steps without prior knowledge of the previous pose. Garon et al. [7,8], formulated the tracking problem exclusively as a learning one, by generating two streams of synthetic RGB-D frame pairs from independent viewpoints and regressing the pose using a CNN. Li et al.[23] initialized a similar CNN architecture using a FlowNet2 [17] backbone and fused its two streams by subtraction. In DeepTAM [39], the training was performed with an Optical flow-based regularization term and the production of multiple heterogenous pose hypotheses was encouraged. Those hypotheses were bootstrapped in the final layer. Last but not least, Deng et al.[4] extended the framework of [33] by combining it with a Rao-Blackwellized Particle Filter [5]. In brief, they randomly sampled 2D bounding boxes to crop RGB images, infer 3D translation probabilities from their dimensions and search for the closest saved rotated sample provided by the Autoencoder of [33]. Then, the particles were weighted according to the orientation probabilities and were prepared for the next sampling iteration.

3 Methodology

3.1 Problem Formulation

Our problem consists in estimating the object pose $\mathbb{P}$, which is usually described as a rigid 3D transformation w.r.t. a fixed coordinate frame, namely an element of the Special Euclidean Lie group in 3D: $SE(3)$. It can be disentangled into two components; a rotation matrix R, which is an element of the Lie Group SO(3) and a translation vector $\mathbf{t} \in \mathbb{R}^3$. However, Brégier et al. [2] proposed a

broader definition for the object pose, which can be considered as a family of rigid transformations, accounting for the ambiguity caused by possible rotational symmetry, noted as G ∈ SO(3). We leverage this augmented mathematical definition for introducing a relaxation to the pose space $\mathscr{C}$ definition:

$$\mathscr{C} = \left\{ \mathbb{P} \quad | \quad \mathbb{P} = \left[\begin{array}{c|c} R \cdot G & \mathbf{t} \\ \hline \mathbf{0}^T & 1 \end{array} \right], \mathbf{t} \in \mathbb{R}^3, R \in \mathrm{SO}(3), G \in \mathrm{SO}(3) \right\}. \tag{1}$$

For example, as stated in [2], the description of the pose of an object with spherical symmetry requires just 3 numbers: $(t_{x,y,z})$, as G can be any instance of SO(3) with the imprinted shape of the object remaining the same. Obviously, for asymmetrical objects, G=$\mathbb{I}_3$.

3.2 Architecture Description

The proposed architecture is depicted in Fig.2. Our CNN inputs two RGB-D frames of size 150 × 150: I(t),$\hat{\mathrm{I}}$(t) (with I(t) being the "Observed" and $\hat{\mathrm{I}}$(t) the "Predicted" one) and regresses an output pose representation $\Delta\mathbf{p} \in \mathbb{R}^9$, with 3 parameters for translation ($\hat{t}_{x,y,z} \in [-1, 1]$) and 6 for rotation. The first two layers of the "Observed" stream are initialized with the weights of a ResNet18[13], pretrained on Imagenet [3], to narrow down the real-synthetic domain adaptation gap, as proposed in [14]. Since Imagenet contains only RGB images, we initialize the weights of the Depth input modality with the average of the weights corresponding to each of the three RGB channels. Contrary to [14], we find beneficial not to freeze those two layers during training. The reason is that we aim to track the pose of the single objects we train on and not to generalize to unseen ones. So, overfitting to that object's features helps the tracker to focus only on distinguishing the pose change. The weights that correspond to the Depth stream are, of course, not frozen in either case. To the output of the second "Observed" layer, we apply spatial attention for foreground extraction and occlusion handling and we add their corresponding output feature maps with the one of the second layer, along with a Residual connection [12] from the first layer. As a next step, we fuse the two streams by concatenating their feature maps and pass this concatenated output through three sequential Fire modules [16], all connected with residual connections [13].

Background and Occlusion Handling: After our first "Observed" Fire layer, our model generates an attention weight map by using a Fire layer dedicated to occlusion handling and foreground extraction, respectively, followed by a 1 × 1 convolution that squeezes the feature map channels to a single one (and normalized by softmax). Our goal is to distil the soft foreground and occlusion segmentation masks from the hard binary ground-truth ones (that we keep from augmenting the object-centric image with random backgrounds and occluders) in order to have their estimations available during the tracker's inference. To this end, we add the two corresponding binary cross entropy losses to our overall loss function. We argue our design choice of using two attention modules, as after experimentation, we found that assigning a clear target to each of the two modules is more beneficial, rather than relying on a single attention layer to resolve both challenges (see Sect.4.3), an observation also reported in [18].

Rotation representation: From a mathematical standpoint, immediate regression of pose parameters [8] with an Euclidean loss is suboptimal: while the translation component belongs to the Euclidean space, the rotation component lies on a non-linear manifold of SO(3). Thus, it is straightforward to model the rotation loss using a Geodesic metric [15,10] on SO(3), i.e. the length, in radians, of the minimal path that connects two of its elements: $\Delta\hat{R}, \Delta R$:

$$L_{Rot}(\Delta\hat{R}, \Delta R_{GT}) = d_{Rot}^{(Geod)}(\Delta\hat{R}, \Delta R_{GT}) = \arccos\Big(\frac{\mathrm{Tr}\left(\Delta\hat{R}^T \cdot \Delta R_{GT}\right) - 1}{2}\Big). \tag{2}$$

In order to minimize the rotation errors due to ambiguities caused by the parameterization choice, we employ the 6D continuous rotation representation that was introduced in [40]: $\Delta\mathbf{r} = (\Delta\mathbf{r_x}^T, \Delta\mathbf{r_y}^T)$, where $\Delta\mathbf{r_{x/y}} \in \mathbb{R}^3$. Given $\Delta\mathbf{r}$, the matrix $\Delta R = (\Delta\mathbf{R_x}, \Delta\mathbf{R_y}, \Delta\mathbf{R_z})^T$ is obtained by:

$$\begin{aligned} \Delta\mathbf{R_x} &= N(\Delta\mathbf{r_x}) \\ \Delta\mathbf{R_y} &= N[\Delta\mathbf{r_y} - (\Delta\mathbf{R_x^T} \cdot \mathbf{r_y}) \cdot \mathbf{\Delta R_x})] \\ \Delta\mathbf{R_z} &= \Delta\mathbf{R_x} \times \Delta\mathbf{R_y} \end{aligned} \tag{3}$$

where $\Delta\mathbf{R_{x/y/z}} \in \mathbb{R}^3$, $N(\cdot) = \frac{(\cdot)}{\|(\cdot)\|}$ is the normalization function. Furthermore, as it has already been discussed in [2], each 3D rotation angle has a different visual imprint regarding each rotation axis. So, we multiply both rotation matrices with an approximately diagonal Inertial Tensor Λ, calculated on the object model's weighted surface and with respect to its center mass, in order to assign a different weight to each rotational component. We note here that since we want that matrix product to still lie in SO(3), we perform a Gramm-Schmidt orthonormalization on the Inertial Tensor Λ before right-multiplying it with each rotation matrix. Finally, we weigh the translation and rotation losses using a pair of learnable weights $\mathbf{v} = [v_1, v_2]^T$ that are trained along with the rest of the network's parameters using Gradient Descent-based optimization, as proposed in [20].

Symmetric Object Handling In the special case of symmetric objects, we disentangle the ambiguities inserted due to this property from the core of rotation estimation. We classify such ambiguities to two distinctive categories: a continuous set of rotational ambiguities (the unweaving of which we incorporate into our loss function) and a discrete set of reflective ambiguities that appear due to our rotation representation choice and are handled heuristically.

Normally, the presence of the RGB input cue would break any symmetry ambiguities as their origin is its Depth counterpart that depicts the object's 3D shape. However, our preliminary experiments have shown that this is only partially true since the tracker places more emphasis to the Depth cue, in general, in its effort to estimate the object's pose. This inclination keeps the ambiguities present during inference and incurs the need to explicitly model this disentanglement in the loss function formulation.

Symmetric Object Handling: Discrete Reflective Symmetry

By replacing the 3D Euler rotational parameter regression of Garon et al.[8], with its 6D continuous counterpart, we face the extra problem of being unable

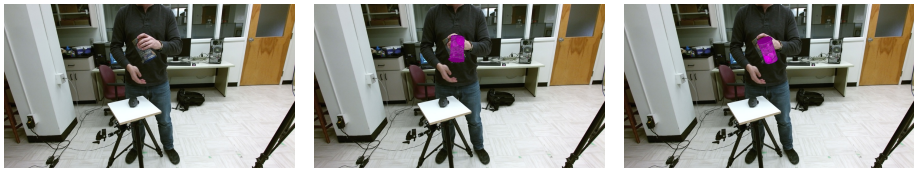

Fig. 3: Comparison of the Rotational Estimation for the 127th frame of the "Hard Interaction Scenario" without/with our proposed reflective symmetry handling algorithm. (a) The ground-truth prediction for frame 127. (b) The rotation estimation of the 126th frame. Rotational error between the prediction and ground truth is small. (c) An erroneous rotational estimation of the tracker for frame 127. The prediction is discarded and replaced by the one of the previous timeframe. The error is kept small and (although larger than in the 126th frame) reflective ambiguity is not propagated to the next timestep via the previous pose feedback rendering.

to constrain the network's rotational output. This has a severely negative effect to objects with reflective symmetry, as there are configurations in which the tracker predicts unacceptable values for one or more rotational components. For example, in Fig. 3.2 we observe that the "Cookie Jar" symmetric object has been turned upside down with respect to both the prediction of the previous timeframe and the ground truth pose . As a result, we end up with adding significant extra errors during the mean rotational error calculation over the total motion length, since that discrepancy is not limited to a single frame, but is accumulated as we proceed to the following frames, due to the temporal nature of the tracker, until it is reset. In order to handle this challenge, we employ the following heuristic algorithm: for each Euler rotational component we calculate the angular distance $d_A^{(o)}(\hat{r}_i(t), \hat{r}_i(t-1))$ (in degrees), between the current and the previous timestep. If a $d_A^{(o)}$ exceeds a certain threshold (here, it is set to 100^o), then the value of this particular Euler angle is set to $\hat{r_i}(t-1)$. Then, we may choose to perform a second (or more) iterative forward CNN pass(es), if the time constraints of our broader application allow so.

Symmetric Object Handling: Continuously Rotational Symmetry In order to form G, we train a batch (of size B_2) of separate Euler angle triplets $\hat{\mathbf{g}} \in \mathbb{R}^3$. At each timestep, one of them is selected and it is converted to a rotation matrix $\hat{G}^*$, which gets right-multiplied with $\hat{\Delta R}$ before being weighted by the parameters of $\Lambda_{(G.S.)}$ (see eq.4). Aiming to select the appropriate parameter from the batch, we train a linear classification layer on top of the first fully connected layer of the tracker. Moreover, we encourage the symmetry triplets to be as uniform as possible by incorporating an appropriate penalty to the overall loss function. To test our approach for the symmetric object case, we used the cylindrical "Cookie Jar" model of [8], the shape of which has only one axis of continuous symmetry. Consequently, we estimate a single rotational symmetry parameter, that of the object-centric z-axis (and keep the rest to 0). On the other hand, in the previous case, we define all three of its axes as axes of reflective symmetry. Before the conversion, that parameter is passed through a tanh function and multiplied by π to constrain its values.

Overall Loss As a result, our overall tracking loss function is formulated as:

$$L_{Track}(\Delta\hat{\mathbb{P}}, \Delta\mathbb{P}) = e^{(-v_1)} \cdot MSE[(\Delta\hat{\mathbf{t}}, \Delta\mathbf{t})] + v_1 + v_2 + \\ + e^{(-v_2)} \cdot arcos\left(\frac{\mathrm{Tr}\left((\Delta\hat{R} \cdot \hat{G}^* \cdot \Lambda_{(G.S.)})^T \cdot (\Delta R \cdot \Lambda_{(G.S.)})\right) - 1}{2}\right) \quad (4)$$

Using a similar external multi-task learnable weighting scheme ($\mathbf{s} = [s_1, s_2, s_3]^T$) as in eq.(4), we combine our primary learning task, pose tracking, with the two auxiliary ones: clutter and occlusion handling. Both $\mathbf{s}$ and $\mathbf{v}$ are initialized to $\mathbf{0}$.

$$Loss = e^{(-s_1)} \cdot L_{Track} + e^{(-s_2)} \cdot L_{Unoccl} + e^{(-s_3)} \cdot L_{Foregr} + s_1 + s_2 + s_3 \quad (5)$$

For objects with continuously rotational symmetry the loss becomes:

$$Loss^{(Symm)} = Loss + e^{(-s_4)}\Big(\frac{1}{B}\sum_{b=1}^{B}\frac{1}{\xi_b}\Big) + s_4, \text{ with} \quad (6)$$

$$\xi_b = \frac{1}{B_2(B_2-1)}\sum_{j=1}^{B_2}\sum_{k\neq j} d_{Rot}^{(Geod)}(\hat{G}_k, \hat{G}_j), \quad (7)$$

The extra term added to the multi-task loss is a penalty that guides the classification layer(s) to select the proper rotational symmetry parameter(s) at each timestep. It encourages the geodesic rotational distances between all pairs of parameters in the batch to be maximum and, thus, ultimately converge to as a uniform distribution as possible. Here, we train B_2=64 such parameters for each continuous rotational component.

3.3 Data Generation and Augmentation

Following [7], for our network (Fig.2), we generate two synthetic RGB-D pairs I(t),$\hat{\mathrm{I}}(t)$, but we alter their sampling strategy using the "Golden Spiral" approach [21], and we modify the augmentation procedure of [7,8] as follows: Firstly, we blend the object image with a background image, sampled from a subset of the SUN3D dataset [37]. We also mimic the procedure of [7,8] in rendering a 3D hand model-occluder on the object frame with probability 60%. A twist we added, is preparing our network for cases of 100% occlusion, by completely covering the object by the occluder for 15% of the occluded subset. Note that both the foreground and unoccluded object binary masks are kept during both of these augmentation procedures. Hence, we can use them as ground truth segmentation signals for clutter extraction and occlusion handling in our auxiliary losses to supervise the corresponding spatial attention maps. We add to the "Observed" frame pair I(t): (i) Gaussian RGB noise, (ii) HSV noise, (iii) blurring (to simulate rapid object movement), (iv) depth downsampling and (v) probabilistic dropout of one of the modalities, all with same parameters as in [8]. With a probability of 50%, we change the image contrast, using parameters $\alpha \sim U(0, 3)$, $\beta \sim U(-50, 50)$ (where U$(\cdot)$ is a uniform distribution) and gamma correction $\gamma \sim U(0, 2)$ with probability 50%, to help generalize over cases of illumination

Object	Attributes				
	Size	Symmetry	Shape	Texture	Distinctive parts
"Dragon"	Medium	No	Complex	Rich	Yes
"Cookie Jar"	Medium	Rotoreflective	Simple	Poor and Repetitive	No
"Dog"	Medium	No	Complex	Almost None	Yes
"Lego"	Small	No	Complex	Rich and Repetitive	No
"Watering Can"	Big	No	Simple	Poor	Yes

Table 1: Characteristics of the five objects we test our approach on. The fact that there are no two identical items validates the generalization capabilities of our tracker.

differences between rendered and sensor generated images. Instead of modelling the noise added to the "Observed" Depth modality with an ad-hoc Gaussian distribution as in [8], we consider the specific properties of Kinect noise [28] and model it with a 3D Gaussian noise (depending on depth and the ground truth object pose), used for simulating the reality gap between synthetic and real images. Its distribution consists of a product of an z-Axial: $n_A \sim \mathcal{N}(0, \sigma_A)$ and two z-Lateral: $n_{L_x} \sim \mathcal{N}(0, \sigma_{L_x})$, $n_{L_y} \sim \mathcal{N}(0, \sigma_{L_y})$ 1D distributions that vary with the object depth z and its angle around the y-axis, θ_y: with standard deviations σ_A,σ_{L_x},σ_{L_y} respectively. The rest of the preprocessing follows [7].

4 Evaluation and Results

4.1 Implementation Details

We use ELU activation functions, a minibatch size of 128, Dropout with probability 0.3, Adam optimizer with decoupled weight decay [26] by a factor $1e^{-5}$, learning rate $1e^{-3}$ and a scheduler with warm restarts [25] every 10 epochs. All network weights with ELU activation function (except those transferred from ResNet18 [13]) are initialized via a uniform K.He [12] scheme, while for all those with a symmetric one we use a corresponding Xavier [9] distribution. Since the Geodesic distance suffers from multiple local minima [6], following [27], we first warm-up the weights, aiming first to minimize the LogCosh[1] loss function for 25 epochs. Then, we train until convergence, minimizing the loss (5). The average training time is 12 hours in a single GeForce 1080 Ti GPU.

4.2 Dataset and Metrics

We test our approach on all the "Interaction" scenarios and the highest percentage "Occlusion" scenarios of [8], which are considered the most challenging. As in [8], we initialize our tracker every 15 frames, and use the same evaluation metrics.

Due to limited computational resources, we produced only 20,000 samples, whose variability covers the pose space adequately enough to verify the validity of our experiments, both for the ablation study and the final experimentation. In the following tables, similarly to [8], we report the mean and standard deviation of our error metrics, as well, as the overall tracking failures (only in the final experimentation section).

	Translational Error(mm)	Rotational Error(degrees)
Garon et al. [8]	34.38 ± 24.65	36.38 ± 36.31
Only occlusion	17.60 ± 10.74	37.10 ± 35.08
Hierarchical clutter & occlusion	14.99 ± **9.89**	39.07 ± 33.22
Parallel clutter & occlusion	**14.35** ± 10.21	**34.28** ± **29.81**

Table 2: Comparison of different attentional foreground/occlusion handling configurations added to the baseline architecture of Garon et al.[8].

	Rotational Error(degrees)
Garon et al. [8]	36.38 ± 36.31
Rotational MSE	46.55 ± 40.88
Geod.	37.69 ± 35.39
Geod.+[40]	14.90 ± 21.76
Geod.+[40]+$\mathcal{A}_{(G.S.)}$	**9.99** ± **13.76**

Table 3: The evolution of the proposed rotation loss, on the baseline architecture of Garon et al. [8] (without our proposed attention modules).

	Translational Error(mm)	Rotational Error(degrees)
Garon et al. [8]	48.58 ± 38.23	36.38 ± 36.31
Steady Weights	11.83 ± 8.94	10.98 ± 16.74
Recursive Batch Standarization [34]	13.97 ± 10.23	14.76 ± 19.24
Learnable Weights [20]	**11.63** ± **8.79**	**8.31** ± **6.76**

Table 4: Comparison of different multi-task weighting schemes.

4.3 Ablation Study

In this section, we discuss our main design choices and we demonstrate quantitative results that led to their selection.

Hierarchy choices for the attention modules Here, we justify the need for both attention modules of our architecture (Fig. 2). We build upon the network proposed by [8] and we firstly introduce a single convolutional attention map just for occlusion handling . Then, we explore the possibility for a seperate attentional weighting of the "Observed" feature map for foreground extraction, prior to the occlusion one, and we, finally, leverage both in parallel and add their resulting maps altogether.

The comparison of **Table 2** establishes not only the need for both attentional modules in our design, but also that the parallel layout is the optimal one. We can observe the effect of parallel connection in Fig.1 as both attentions present sharp peaks. We can, also, observe a visual tradeoff between the parallel attentions: while the object is not occluded (either in steady state or when moving), the module responsible for foreground extraction is highlighted more intensely than the occlusion one. As the object gets more and more covered by the user's hand, the focus gradually shifts to the module responsible for occlusion handling. Note that this is not an ability we explicitly train our network to obtain, but rather a side effect of our approach, which fits our intuitive understanding of cognitive visual tracking. Moreover, although our supervising signals are uniform, both attentional modules learn to highlight specific keypoints of interest during the tracker's inference.

We, also, observe another interesting property: the tracker learns to handle self-occlusion patterns as well. For example, for the "Dragon" object of Fig.1, it learns to ignore the one wing when it is in front and focuses on the other one, at

	Translational Error(mm)	Rotational Error(*degrees*)	$d_A^{(o)}(\hat{r}_z, r_z^{GT})$
Garon et al.[8]	10.75 ± 6.89	23.53 ± 18.85	10.60 ± 38.90
Ours	10.87 ± 8.14	20.55 ± 18.06	4.60 ± 35.42
Ours+Unique,frozen learnable symmetry parameter	11.38 ± 8.94	10.98 ± 16.74	2.98 ± 25.07
Ours+Regression of learnable symmetry parameter	13.03 ± 6.88	17.25 ± 12.40	2.84 ± 24.95
Ours+Mean of a batch of frozen, learnable symmetry parameters	11.98 ± 9.23	13.84 ± 11.87	2.16 ± 29.25
Ours+Optimal selection out of a batch of frozen, learnable symmetry parameters	**10.43 ± 6.63**	**9.57 ± 10.01**	**2.07 ± 24.52**

Table 5: Comparison of different selection methods for the continuously rotational symmetry parameter.

the back, if its appearance is more distinctive of the pose. The same can be said for the legs of the "Dog" model of Fig.6. This is a clue that this module has, indeed, learned the concept of occlusions and has not overfitted to the shape of the user's hand, the occluder model that was used for training.

Contributions of the rotation Loss components We demonstrate the value of every component included in our rotation loss (leaving symmetries temporarily out of study), by: (i) regressing only the rotational parameters with the baseline architecture of [8], (ii), replacing the MSE loss with the Geodesic one, (iii), replacing the rotation parameterization of [8] with the continuous one of (3), and, (iv) including the Inertial Tensor weighting of each rotational component.

Table 3 indicates the value that translation estimation brings to rotation estimation, as when the former's regression is excluded, the latter's performance decreases. Moreover, **Table 3** justifies our progressive design selections in formulating our rotation loss, as with the addition of each ambiguity modelling, the 3D rotation error decreases, starting from $46.55^o \pm 40.88^o$ and reaching $9.99^o \pm 13.76^o$.

Weighting the Multi-Task Loss Here, we explore various weighting schemes of the multiple loss functions of our approach, both the main and the auxiliary ones. Our first approach is the crude addition of the tracking and the two Binary Cross Entropy losses . A second one is standarizing the three losses by subtracting their batchwise means and dividing by their batchwise standard deviations, that we calculate using the Welford algorithm[34]. Lastly, we consider the learnable weighting strategy that we, ultimately, utilize . In quantitative comparison of **Table4** the scheme of [20] emerges as the clear favourite.

Comparison of Continuously Rotational Symmetry Regression methods ("Cookie Jar" model) In our effort to disentangle the rotation estimation and the continuously rotational symmetries, we try different configurations for optimally choosing the appropriate parameter(s): (i) learning a unique symmetry parameter over all possible pose changes in the training set and keeping it frozen during inference, (ii) regressing a different one per pose pair, (iii) learning a batch of them and taking their average during inference and, ultimately, (iv) selecting the optimal from the aforementioned batch using an appropriate classification layer while encouraging the values of this batch to be as uniform as possible, at the same time. This freedom of ours resides from the fact that the

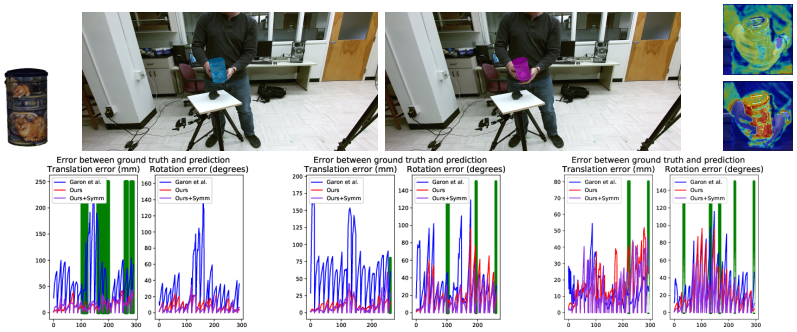

Fig. 4: Comparison of the SoA[8] *(light blue)* and our *(pink)* approaches for the "Cookie Jar" in 3 scenarios: "Translation Only", "Rotation Only" and "Full Interaction".

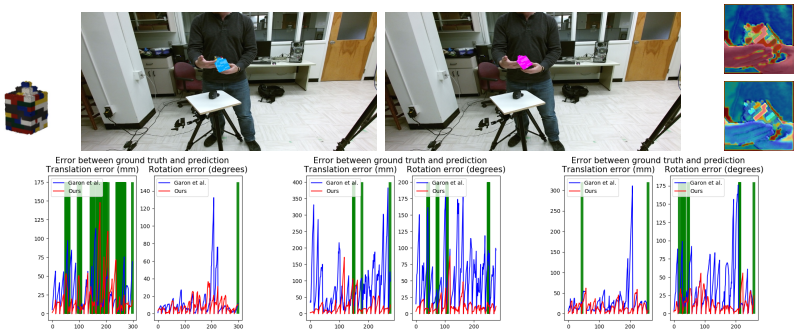

Fig. 5: Comparison of the SoA[8] *(light blue)* and our *(pink)* approaches for the "Lego" in 3 scenarios: "Translation Only",'Full' and "Hard Interaction".

minimization of the tracking loss w.r.t. the symmetry matrix $\hat{G}^*$ (see [2]) does not explicitly impose a global-solution constraint. This time, the comparison is done w.r.t. the full approach of Sect.3 that does not account for symmetries and, apart from the classic 3D translational and rotational errors, we, also, report the Euler angle error for the z-component. As we can see in **Table 5**, our proposed approach is the "golden medium" between accuracy and robustness since both its mean and standard deviation are the lowest across all metrics.

4.4 Experimental Results

According to our ablation study, we proceed to merge our parallel attention modules with the Geodesic rotation loss of eq.(4), along with the remaining elements of Sect.3. We evaluate our method on five objects of dataset [8]: the "Dragon", the "Cookie Jar", the "Dog", the "Lego" block and the "Watering Can", aiming for maximum variability (see **Table 1**). In Fig. 1-7, we plot the 3D translational and rotational errors in three randomly selected scenarios for each object both for the SoA and our tracker.

Evidently, the object most benefited by our methodological improvements is the "Dragon". Since its geometry is the most complex, its texture is rich and it has several distinctive parts that stand out of the user's grip, both our geometric modelling and the parallel attention modules find their best application in this

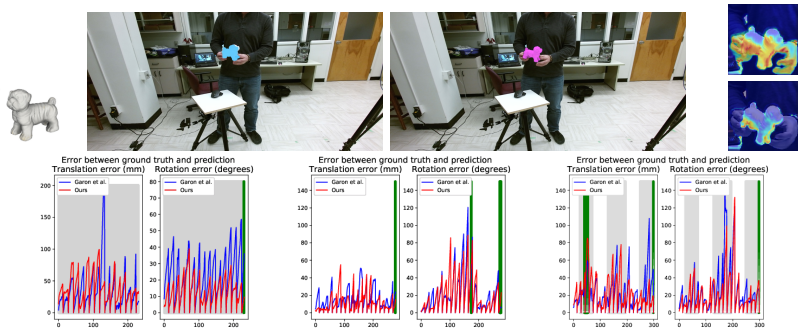

Fig. 6: Comparison of the SoA[8] *(light blue)* and our *(pink)* approaches for the "Dog" in 3 scenarios: "75% Vertical Occlusion", "Rotation Only" and "Hard Interaction".

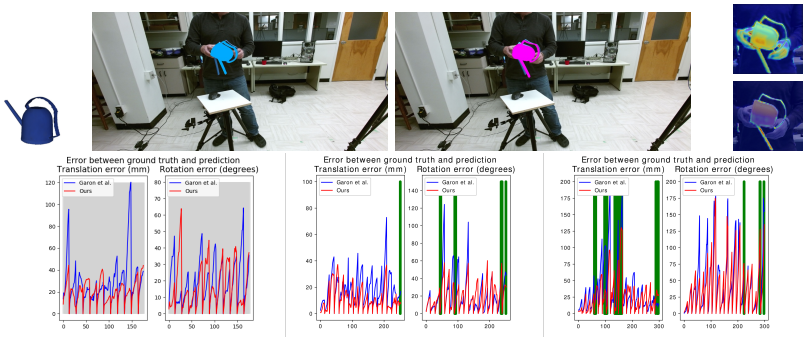

Fig. 7: Comparison of the SoA[8] *(light blue)* and our *(pink)* approaches for the "Watering Can" in 3 scenarios:"75% Horizontal Occlusion", "Rotation Only" and "Full Interaction".

case. When the user's hand occludes parts of the "Dragon", the attention shifts to its body parts of interest that stand out of the grip, like its neck, wings or tail (Fig.1). For the symmetric "Cookie Jar", the differences between our method and the baseline are lower. The attentions' effect is less prominent here since this model is of simpler, symmetric shape and poorer texture. This replaces the distinctive clues of the dragon case with ambiguities, denying the corresponding modules of the ability to easily identify the pose. Alongside the "Lego" model, they make the most out of the reflective symmetry handling algorithm as they avoid large abrupt errors that propagate to future frames. Although our CNN primarily focuses on the object's shape, appearance seems to play a significant role in its predictions, as well, since the errors of the "Dog" and the "Watering Can" models, the less textured ones, decrease more mildly. The foreground attention map aids disentangling the "Dog" model from its background, a table of the same color, in the "75% Occlusion" scenarios. On the other hand, for the "Watering Can", the most ambiguities are presented when viewpoint-induced symmetries appear, with the effects of our modelling declining in this case.

The accuracy of our approach exceeds that of the SoA [8], across all objects and in almost all scenarios, especially for 3D rotations. According to **Table6**, both errors are generally lower (in terms of both mean and standard deviation)

Approach	75% Horizontal Occlusion			75% Vertical Occlusion		
	Translational Error(*mm*)	Rotational Error(*degrees*)	Fails	Translational Error(*mm*)	Rotational(*degrees*)	Fails
Garon et al.[8] ('Dragon)	16.02 ± **8.42**	18.35 ± 11.71	13	18.20 ± 11.81	14.66 ± 12.98	13
Ours("Dragon")	**12.68** ± 11.49	**13.00 ± 9.14**	**10**	**12.87 ± 10.49**	**13.14 ± 8.85**	**8**
Garon et al.[8]("Cookie Jar")	21.27 ± 9.74	21.90 ± 13.97	17	20.77 ± **6.88**	24.86 ± 13.64	20
Ours("Cookie Jar")	9.51 ± 4.17	15.48 ± 9.50	15	20.97 ± 7.32	16.14 ± 10.06	15
Ours+Symm.("Cookie Jar")	**6.37 ± 2.14**	**7.22 ± 3.97**	**11**	**19.01** ± 7.53	**13.00 ± 7.49**	**14**
Garon et al.[8]("Dog")	37.96 ± 23.39	47.94 ± 31.55	**21**	**32.84** ± 34.07	22.44 ± 13.60	21
Ours("Dog")	**24.43 ± 18.92**	**17.24 ± 12.41**	25	36.53 ± **22.39**	**12.67 ± 7.95**	**20**
Garon et al.[8]("Lego")	**68.25** ± 46.97	40.04 ± 47.37	28	40.04 ± 47.37	35.30 ± 31.32	20
Ours("Lego")	72.04 ± **34.10**	**18.41 ± 13.84**	**28**	**12.92 ± 5.73**	**12.92 ± 9.02**	**20**
Garon et al.[8]("Watering Can")	21.59 ± 11.32	23.99 ± **16.95**	14	32.76 ± 24.12	26.74 ± 19.05	18
Ours("Watering Can")	**20.71 ± 10.24**	**17.00** ± 18.99	**13**	**17.66 ± 17.95**	**13.46 ± 10.43**	**12**

Approach	Translation Interaction			Rotation Interaction		
	Translational Error(*mm*)	Rotational Error(*degrees*)	Fails	Translational Error(*mm*)	Rotational(*degrees*)	Fails
Garon et al.[8] ("Dragon")	41.60 ± 39.92	11.55 ± 15.58	15	23.86 ± 17.44	27.21 ± 22.40	15
Ours("Dragon")	**11.05 ± 8.20**	**3.55 ± 2.27**	**1**	**9.37 ± 6.07**	**7.86 ± 6.69**	**2**
Garon et al.[8]("Cookie Jar")	20.43 ± 25.44	17.19 ± 12.99	16	10.75 ± **5.89**	23.53 ± 18.85	19
Ours("Cookie Jar")	8.64 ± 8.23	8.31 ± 5.97	5	10.87 ± 8.14	20.55 ± 18.06	16
Ours+Symm.("Cookie Jar")	**8.09 ± 7.67**	**5.83 ± 5.50**	**3**	**9.98** ± 10.63	**13.84 ± 11.87**	**16**
Garon et al.[8]("Dog")	58.87 ± 71.86	16.42 ± 13.51	20	11.16 ± 10.28	**20.00** ± 21.31	17
Ours("Dog")	**21.64 ± 22.78**	**9.27 ± 8.03**	**14**	**10.68 ± 7.53**	20.07 ± **19.29**	**17**
Garon et al.[8]("Lego")	27.90 ± **23.53**	11.89 ± 18.50	29	16.42 ± 10.90	17.83 ± 15.90	32
Ours("Lego")	**22.66** ± 24.58	**9.08 ± 7.60**	**12**	**10.13 ± 6.79**	**7.22 ± 4.55**	**4**
Garon et al.[8]("Watering Can")	24.95 ± 42.91	13.26 ± 11.34	16	13.14 ± **8.99**	22.19 ± 25.93	15
Ours("Watering Can")	**24.30 ± 21.51**	**8.79 ± 6.35**	**16**	**12.22** ± 9.46	**18.66 ± 15.51**	**15**

Apporach	Full Interaction			Hard Interaction		
	Translational Error(*mm*)	Rotational Error(*degrees*)	Fails	Translational Error(*mm*)	Rotational(*degrees*)	Fails
Garon et al.[8] ("Dragon")	35.23 ± 31.97	34.98 ± 29.46	18	34.38 ± 24.65	36.38 ± 36.31	17
Ours("Dragon")	**10.31 ± 8.66**	**6.40 ± 4.52**	**1**	**11.63 ± 8.79**	**8.31 ± 6.76**	**2**
Garon et al.[8]("Cookie Jar")	**13.06 ± 9.35**	31.78 ± 23.78	24	15.78 ± 10.43	24.29 ± 20.84	15
Ours("Cookie Jar")	17.03 ± 11.94	22.24 ± 20.86	21	15.29 ± 16.06	16.73 ± 14.79	11
Ours+Symm.("Cookie Jar")	14.63 ± 11.19	**15.71 ± 13.80**	**21**	**14.96 ± 9.06**	**15.00 ± 13.20**	**8**
Garon et al.[8]("Dog")	37.73 ± 42.32	**20.77 ± 19.66**	23	23.95 ± 38.86	24.38 ± 26.39	20
Ours("Dog")	**24.88 ± 35.85**	28.52 ± 25.38	**20**	**19.32 ± 15.97**	**19.72 ± 20.17**	**19**
Garon et al.[8]("Lego")	30.96 ± 31.44	22.10 ± 20.20	20	30.71 ± 42.62	36.38 ± 34.99	20
Ours("Lego")	**23.58 ± 27.73**	**11.80 ± 12.28**	**13**	**16.47 ± 12.95**	**14.29 ± 11.68**	**11**
Garon et al.[8]("Watering Can")	33.76 ± 37.62	40.16 ± 35.90	26	28.31 ± 19.49	23.04 ± 24.27	28
Ours("Watering Can")	**19.82 ± 19.98**	**28.76 ± 30.27**	**26**	**18.03 ± 14.99**	**19.57 ± 17.47**	**23**

Table 6: 3D Translational and Rotational errors and overall tracking failures in six different scenarios for five employed objects.

and our tracker fails equally or less often. It presents aggravated errors in fast object motions more rarely than [8] and handles both static and dynamic high-percentage occlusion patterns better. Also, it not only keeps track of the object's 3D position under severe occlusions, but extends this property to 3D rotations, as well. Although more computationally intense than [8], it runs in 40 fps.

5 Conclusion

In this work, we propose a CNN for fast and accurate single object pose tracking. We perform explicitly modular design of clutter and occlusion handling and we account for the geometrical properties of both the pose space and the object model during training. As a result, we reduce both SoA pose errors by an average of 34.03% for translation and 40.01% for rotation for a variety of objects with different properties. Our tracker exceeds the SoA performance in challenging scenarios with high percentage occlusion patterns and rapid movement and we gain an intuitive understanding of our artificial tracking mechanism.

Acknowledgements

This research has been co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH CREATE INNOVATE (project code: T1EDK-01248, acronym: i-Walk).

References

1. Belagiannis, V., Rupprecht, C., Carneiro, G., Navab, N.: Robust optimization for deep regression. In: Proceedings of the IEEE international conference on computer vision. pp. 2830–2838 (2015)
2. Brégier, R., Devernay, F., Leyrit, L., Crowley, J.L.: Defining the pose of any 3d rigid object and an associated distance. Int. J. of Comp. Vision (IJCV) **126**(6), 571–596 (2018)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 248–255. Ieee (2009)
4. 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)
5. Doucet, A., De Freitas, N., Murphy, K., Russell, S.: Rao-blackwellised particle filtering for dynamic bayesian networks. arXiv preprint arXiv:1301.3853 (2013)
6. Fletcher, T.: Terse notes on riemannian geometry (2010)
7. Garon, M., Lalonde, J.F.: Deep 6-dof tracking. IEEE transactions on visualization and computer graphics **23**(11), 2410–2418 (2017)
8. Garon, M., Laurendeau, D., Lalonde, J.F.: A framework for evaluating 6-dof object trackers. In: Proc. European Conf. on Computer Vision (ECCV). pp. 582–597 (2018)
9. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256 (2010)
10. Hartley, R., Trumpf, J., Dai, Y., Li, H.: Rotation averaging. Int. J. of Comp. Vision (IJCV) **103**(3), 267–305 (2013)
11. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proc. IEEE Int. Conf. on Computer Vision (ICCV). pp. 2961–2969 (2017)
12. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proc. IEEE Int. Conf. on Computer Vision (ICCV). pp. 1026–1034 (2015)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
14. Hinterstoisser, S., Lepetit, V., Wohlhart, P., Konolige, K.: On pre-trained image features and synthetic images for deep learning. In: Proc. European Conf. on Computer Vision (ECCV). pp. 0–0 (2018)
15. Huynh, D.Q.: Metrics for 3d rotations: Comparison and analysis. J. Math. Imaging and Vision **35**(2), 155–164 (2009)
16. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size. arXiv:1602.07360 (2016)
17. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: Evolution of optical flow estimation with deep networks. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 2462–2470 (2017)
18. Jafari, O.H., Mustikovela, S.K., Pertsch, K., Brachmann, E., Rother, C.: ipose: instance-aware 6d pose estimation of partly occluded objects. In: Asian Conference on Computer Vision. pp. 477–492. Springer (2018)

19. Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In: Proc. IEEE Int. Conf. on Computer Vision (ICCV). pp. 1521–1529 (2017)
20. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 7482–7491 (2018)
21. Leopardi, P.C.: Distributing points on the sphere: partitions, separation, quadrature and energy. Ph.D. thesis, University of New South Wales, Sydney, Australia (2007)
22. Lepetit, V., Moreno-Noguer, F., Fua, P.: Epnp: An accurate o(n) solution to the pnp problem. Int. J. of Comp. Vision (IJCV) **81** (02 2009). https://doi.org/10.1007/s11263-008-0152-6
23. Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: Deepim: Deep iterative matching for 6d pose estimation. In: Proc. European Conf. on Computer Vision (ECCV). pp. 683–698 (2018)
24. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Proc. European Conf. on Computer Vision (ECCV). pp. 21–37. Springer (2016)
25. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 (2016)
26. Loshchilov, I., Hutter, F.: Fixing weight decay regularization in adam. arXiv preprint arXiv:1711.05101 (2017)
27. Mahendran, S., Ali, H., Vidal, R.: 3d pose regression using convolutional neural networks. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 2174–2182 (2017)
28. Nguyen, C.V., Izadi, S., Lovell, D.: Modeling kinect sensor noise for improved 3d reconstruction and tracking. In: 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission. pp. 524–530. IEEE (2012)
29. Park, K., Patten, T., Vincze, M.: Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 7668–7677 (2019)
30. Pavlakos, G., Zhou, X., Chan, A., Derpanis, K.G., Daniilidis, K.: 6-dof object pose from semantic keypoints. In: 2017 IEEE international conference on robotics and automation (ICRA). pp. 2011–2018. IEEE (2017)
31. 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)
32. Segal, A., Haehnel, D., Thrun, S.: Generalized-icp. In: Robotics: science and systems. vol. 2, p. 435. Seattle, WA (2009)
33. Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R.: Implicit 3d orientation learning for 6d object detection from rgb images. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 699–715 (2018)
34. Welford, B.: Note on a method for calculating corrected sums of squares and products. Technometrics **4**(3), 419–420 (1962)
35. 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)
36. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199 (2017)

37. Xiao, J., Owens, A., Torralba, A.: Sun3d: A database of big spaces reconstructed using sfm and object labels. In: Proc. IEEE Int. Conf. on Computer Vision (ICCV). pp. 1625–1632 (2013)
38. Zakharov, S., Shugurov, I., Ilic, S.: Dpod: Dense 6d pose object detector in rgb images. arXiv preprint arXiv:1902.11020 (2019)
39. Zhou, H., Ummenhofer, B., Brox, T.: Deeptam: Deep tracking and mapping. In: Proc. European Conf. on Computer Vision (ECCV). pp. 822–838 (2018)
40. Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the continuity of rotation representations in neural networks. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 5745–5753 (2019)