

Article

# DeepHMap++: Combined Projection Grouping and Correspondence Learning for Full DoF Pose Estimation

Mingliang Fu <sup>1,2,3,\*</sup>  and Weijia Zhou <sup>1,2</sup>

<sup>1</sup> State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China; zwj@sia.cn

<sup>2</sup> Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110016, China

<sup>3</sup> University of Chinese Academy of Sciences, Beijing 100049, China

\* Correspondence: fumingliang@sia.cn; Tel.: +86-024-2397-0188

Received: 24 January 2019; Accepted: 25 February 2019; Published: 28 February 2019

**Abstract:** In recent years, estimating the 6D pose of object instances with convolutional neural network (CNN) has received considerable attention. Depending on whether intermediate cues are used, the relevant literature can be roughly divided into two broad categories: direct methods and two-stage pipelines. For the latter, intermediate cues, such as 3D object coordinates, semantic keypoints, or virtual control points instead of pose parameters are regressed by CNN in the first stage. Object pose can then be solved by correspondence constraints constructed with these intermediate cues. In this paper, we focus on the postprocessing of a two-stage pipeline and propose to combine two learning concepts for estimating object pose under challenging scenes: projection grouping on one side, and correspondence learning on the other. We firstly employ a local-patch based method to predict projection heatmaps which denote the confidence distribution of projection of 3D bounding box's corners. A projection grouping module is then proposed to remove redundant local maxima from each layer of heatmaps. Instead of directly feeding 2D–3D correspondences to the perspective-n-point (PnP) algorithm, multiple correspondence hypotheses are sampled from local maxima and its corresponding neighborhood and ranked by a correspondence–evaluation network. Finally, correspondences with higher confidence are selected to determine object pose. Extensive experiments on three public datasets demonstrate that the proposed framework outperforms several state of the art methods.

**Keywords:** 6D pose estimation; partial occlusion; projection grouping; correspondence evaluation

## 1. Introduction

Estimating the full degree-of-freedom (DoF) pose of a rigid object, meaning 3D translation and 3D orientation from a single frame is an important topic in the realm of computer vision. A huge number of approaches have been proposed to address applications in the domain such as robotics, augmented reality and medical navigation [1]. Real time is a key indicator for almost all applications from the three fields above. One line of solutions is modeling objects into a sparse set of feature points [2]. For well-textured objects, this problem has been well addressed by constructing correspondence constraints between the prior model and scene image [2,3]. However, both robustness and accuracy continue to be critical issues that limit existing methods under challenging scenarios [4]. Thus, many researchers have recently begun to employ convolutional neural network (CNN) or ensemble learning [5] to address these issues.

Depending on whether intermediate cues such as 3D object coordinate [6], projection of virtual control points [7], or semantic keypoints [8] are used, related approaches can be roughly divided into

two categories: direct methods and two-stage pipelines. For a typical two-stage pipeline, intermediate cue is prepared in the first stage and pose parameters is computed by these intermediate cues in the back-end. Throughout the rest of this paper, the front-end and back-end correspond to the first and second phase of a two-stage pipeline, respectively. Instead of predicting directly the full DoF pose, Brachmann et al. [6] first compute 3D object coordinates and the confidence map of scene pixels with random forests. Dense correspondences then are transferred instantly to a random sample consensus (RANSAC) based optimization step. Encoding local feature of the input image makes 3D object coordinates inherently robust to partial occlusion and achieves top-level results on the Occluded LineMOD dataset [6]. However, Brachmann et al. [6] didn't specially consider the case of symmetrical objects [9]. Sparsity is another important requirement to ensure the robustness to heavy occlusion. In contrast to dense object coordinates [6], both projections of virtual control points [10] and bounding box's corners [7] are formally sparse. Oberweger et al. [11] taken into account both sparsity and locality, and upgraded the robustness of BB8 [7] (8 corners of the bounding box) by predicting projection heatmaps from random local patches. For simplicity, the method reported in Ref. [11] is named DeepHMap. As a part-based method, DeepHMap [11] lifts the robustness to occlusion to a new level with simple local patches. However, predicted heatmaps always encounter multiple local maxima due to the absence of global information. Oberweger et al. [11] directly select the global maxima to construct correspondence constraints without considering the rationality of corner projection. Compared with a wide range of intermediate cues, the postprocessing corresponding to the second stage is seldomly noticed. The studies [12–14] are three of the few postprocessing stages that begin to divert attention to the back-end, all of which are less portable because of the depth of customization.

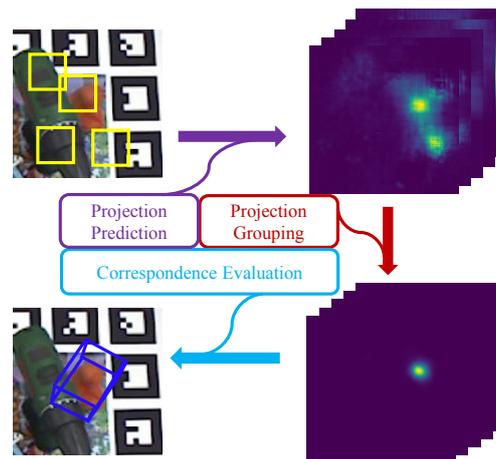
Motivated by the above analysis, we focus on the back-end of a two-stage pipeline for ensuring both accuracy and robustness in this paper. Given the simple yet efficient strategy of baseline [11], we follow the same line to achieve projection heatmaps of 3D bounding boxes' corners (BBCs). For raw merged heatmaps from the baseline [11], a good postprocessing method should have the following features: (1) the postprocessing can be seamlessly integrated with the front-end network. That is, we do not have to spend extra effort to connect these two parts; (2) the back-end should be efficient enough, and introducing a heavy computational cost in exchange for improved accuracy is not advisable; and (3) unreasonable projection distribution on a single-layer of heatmaps should be properly excluded. To this end, we present a two-stage approach as depicted in Figure 1. The proposed method consists of three parts, projection prediction, projection grouping and correspondence evaluation. A simple projection grouping module is designed firstly to learn spatial correlation of projection of different BBCs. Thus, unreasonable local maxima can be removed by geometric constraints learned with this projection grouping module and each layer of the filtered heatmaps contains only one peak. For each layer of heatmap, each pixel stores the corresponding confidence of projection distribution. In fact, current projection predictions are still biased against the ground truth. Multiple correspondence hypotheses thus are sampled from both the only local maxima and its corresponding neighborhood instead of feeding directly 2D–3D correspondences to the perspective-n-point (PnP) method [15]. Similar hypothesis sampling can be found in Ref. [16]. Instead of random hypothesis selection or iterative refinement [13], we delegate all of these to a correspondence evaluation network [17]. Finally, correspondences with higher confidence are chosen to calculate object pose. For brevity, we name our method DeepHMap++ in text below.

In summary, the main contributions of this paper can be concluded as the following:

1. We present a simple yet efficient projection grouping module for removing fake local maxima in each layer of projection heatmaps. The projection grouping module learns correlation constraints among projections of different BBCs and to select the optimal projection.
2. In order to suppress different jitters during inference, multiple correspondence hypotheses are randomly sampled from local maxima and its corresponding neighborhood and ranked by a correspondence–evaluation network.

- We show the effectiveness of projection grouping and corresponding evaluation, and the corresponding two-stage pipeline achieves state-of-the-art performance on public benchmarks including LineMod dataset [18], Occluded LineMOD dataset [6], and YCB-Video dataset [19].

The rest of the paper is structured as follows: an overview of the related works is provided in Section 2. Section 3 describes the complete pipeline. Extensive evaluations and comparisons with several state-of-the-art baselines are demonstrated in Section 4. Conclusions and future work are presented in Section 5.



**Figure 1.** Overview of the proposed two-stage approach for recovering the 6D pose. For any input patch (yellow box), its corresponding output from projection prediction module consists of projection heatmaps with eight layers.

## 2. Related Works

From the early feature-based approaches [2,3] to the recent methods based on machine learning, the field of pose estimation for 3D rigid objects has accumulated a vast range of solutions. In this section, we confine ourselves to the field of 6D pose estimation in single frame via learning. As mentioned in a previous section, related works on 6D pose estimation can be roughly divided into two groups: direct methods [20–32] and two-stage pipelines [4,6–8,10,11,13,14,19,33–39].

### 2.1. Direct Methods

For direct methods, full DoF poses are directly encoded in both learning and inference. Attracted by the efficient LineMOD [18] template, Tejani et al. [20] proposed a latent-class based hough forest that employs a part-based version to improve the robustness to partial occlusion and clutter. Instead of random forest, a convolutional auto-encoder [31] is trained from local view patches and generalizes well both on seen and unseen objects. A similar extension of local patch based regression can be found in Ref. [32]. Wohlhart et al. [21] presented a novel learning based descriptor mapping the object categories and viewpoints to Euclidean space. Thus, a large Euclidean distance between descriptors means different category attributes and distance of descriptors in Euclidean space is directly related to the difference between different views. More aggressively, a pose guided feature [22,30] is designed to learn exact pose differences.

One-shot based 6D pose estimation has been recently frequently addressed in the literature. In Ref. [23], a fully connected auto-encoder is employed to learn latent features. Significant progress in visual object recognition and detection has been made with deep learning. Therefore, many scholars have begun to predict pose parameters in one branch of deep neural networks. Following the state-of-the-art object detector, Mask R-CNN [40], a multi-task learning network [27] with a pose branch is demonstrated. Similar end-to-end fashion for pose estimation can also be found in

Refs. [26,28]. To reduce the reliance on data annotations, an implicit orientation learning [29] is proposed via learning from samples processed by an augmented autoencoder. Mitash et al. [25] presented a comprehensive framework for full DoF pose estimation via Monte Carlo tree search, which completely eliminates a time-consuming labeling step. More recently, a multi-view and multi-class framework [24] demonstrates impressive 6D pose estimation via a multi-class representation of pose space.

## 2.2. Two-Stage Pipeline

The biggest difference between direct method and two-stage pipeline is whether to use intermediate cues or not. A common intermediate cue is segmented point cloud [33] in a pick-and-place system. 6D pose of a rigid object instance is achieved by aligning the segmented point cloud with a pre-scanned 3D model. 3D object coordinate [6] is another flexible intermediate cue, which has been proven to be very efficient for 6D pose estimation [6,8,35,38,41] and camera localization [16]. To cope with a multi-object case, 3D object coordinates together with object labels [34] are jointly employed intermediate cues.

Holistic methods [14,19,36] globally formulate the pose detection issue and feed directly the entire scene image into the regression network. What we want to highlight here is SSD-6D (SSD denotes single shot detector) [36], which originally constructs a 6D hypothesis from 2D bounding boxes. In contrast to 3D object coordinates, a 2D bounding box is an implicit intermediate cue that doesn't explicitly contain 3D information. After significant progress [42] has been made in the face of conventional clutter scenes [18], researchers begin to shift their attention to more challenging scenes under severe occlusion [6]. Compared with the part-based approach [6], a holistic method such as SSD-6D is more likely to be disturbed by the foreground occlusion when constructing the mapping to pose space.

Crivellaro et al. [10] put a novel intermediate cue, virtual control points, into our view. 3D pose of an object part is represented as projections of virtual control points, making it possible to handle poorly textured objects under partial occlusion and heavy clutter. To avoid manually selecting parts of a special object like Crivellaro et al. [10], more general-purpose virtual control points, BBCs [7] are utilized to construct 2D–3D correspondences. Following the same principle of BB8 [7], Oberweger et al. [11] proposed training CNN from random local patches and achieved state-of-the-art performance. A similar part loss [43] or part response [44] based method also shows strong robustness to occlusion in other areas such as face detection. In multi-branch networks [19,39], segmentation mask, 2D bounding box, and object's center are frequently employed as intermediate cues. Different from virtual control points mentioned above, semantic keypoints [37] have also been proved to be an effective intermediate cue. Unfortunately, time-consuming auto-extracting of semantic keypoints hinders its real-time applications.

The following postprocessing is equally important after the acquisition of intermediate cues. After achieving 3D object coordinates using random forests in the first stage, a novel pose agent [13] is designed to repeatedly refine pose hypotheses. This is the only reinforcement learning based example that we can find in the back-end. In the case of a differentiable RANSAC (DSAC) based pipeline [16], finite differences used in refinement gradients lead to high gradient variance during the end-to-end learning. To address the remaining issues in DSAC, a fully differentiable backend [12] is proposed for camera localization. Compared with intermediate cues in the front-end, research about postprocessing is still relatively deficient.

## 3. Methods

According to the definition of the preceding statement, our method belongs to a typical two-stage method. In the front-end, projection heatmaps of 3D BBCs are predicted by the tutorial described in DeepHMap [11]. The core task of the paper is to design a comprehensive postprocessing in

the back-end. Our proposed postprocessing consists of two modules: projection grouping and correspondence learning based hypothesis selection. We describe each necessary step in this section.

### 3.1. Local Patch Based Heatmap Prediction

DeepHMap [11] uses an asymmetric hourglass network for predicting projection heatmaps, which takes a random local patch with size of  $32 \times 32$  as input and produces corresponding predicted heatmaps with size of  $128 \times 128$ . Different from direct projection prediction of BBCs with a holistic patch in BB8 [7], DeepHMap outputs projection heatmaps that denote a confidence distribution of projection. Compared with projection heatmaps, direct pose regression is a more demanding task. Different predicted heatmaps from random local patches are then merged via simple averaging, which constantly produces multiple local maxima in single channel of heatmaps. A more flexible strategy instead of the global maxima [11] is adopted and described in detail in the below subsection.

### 3.2. Projection Grouping

For DeepHMap [11], the key to improve the robustness to heavy occlusion is to feed random local patches instead of holistic objects of interest to CNN. However, local patches mean that the correlation between different parts of a special object is ignored. During the inference, each sample in the minibatch predicts projection heatmaps according to its own content. In each channel of the merged heatmaps, multiple local maxima can be frequently found. Oberweger et al. [11] select the global maxima to eliminate this ambiguity. However, the global maxima is not always the optimal choice.

To solve these ambiguities more thoroughly, we propose a simple projection grouping module to guide the projection selection. For projection distribution on a single channel of heatmaps, the rationality of its location can be evaluated by constraints from two aspects: correlation constraints among projections on different channels on one side, and correspondence constraints between 3D BBCs and their corresponding 2D projections on the other. Next, we elaborate on the design process of the network architecture with considering correlation constraints, and correspondence constraints are fused in subsequent correspondence evaluation step. The first difficulty we have to face is that the number of local maxima on each channel of the heatmaps is always in dynamic change. Each channel of the merged heatmaps may contain projection clusters ranging in number from zero to many. For local patches from background or occlusion areas, heatmaps may not contain peaks. Before detailing more design details, we first revisit the strategy employed in DeepHMap. In particular, let  $X = \{l_1, l_2, \dots, l_8\}$  represents predicted heatmaps consisting of eight channels corresponding to different BBCs. In order to get a group of projections from different heatmap channels, Oberweger et al. [11] consistently choose the global maxima. This simple strategy can be written as:

$$\forall i, 1 \leq i \leq 8, y_i = \max(l_i), \quad (1)$$

where  $\max(\cdot)$  is a function that takes the global maxima from a single-channel heatmap, the output  $y_i$  denotes the  $i$ th channel of filtered heatmaps and contains the predicted projection of the corresponding BBC. Obviously, the above mentioned projection grouping described in Equation (1) is carried out separately on a layer-by-layer fashion.

To fuse correlation constraints mentioned above, a simple fully connected network with residual architecture is employed to learn different projection cases. The task of projection grouping becomes constructing a mapping  $f$  with learned parameters  $\psi$ , such that

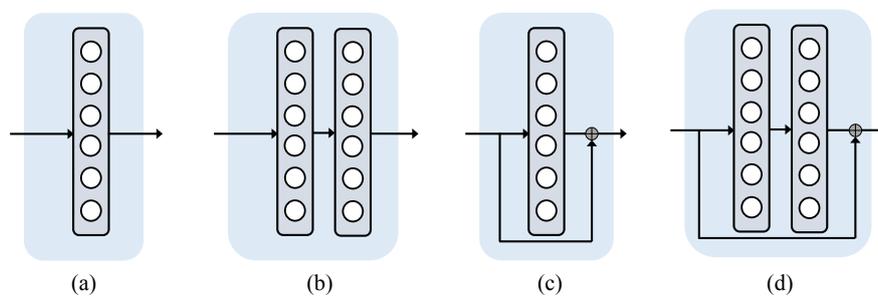
$$\forall i, 1 \leq i \leq 8, y_i = f(X, \psi) + l_i. \quad (2)$$

Compared with Equation (1), the strategy given in Equation (2) takes the correlation constraints among different channels into account. As shown in Figure 2, the projection grouping module takes merged heatmaps adjusted by the spatial transformation layer [45] as input. For the input patch  $p_j$  of a minibatch consisting of  $N_{batch}$  batches, let  $Y_j = [y_1^j, y_2^j, \dots, y_8^j]$  and  $O_j = [o_1^j, o_2^j, \dots, o_8^j]$  represent the

predicted heatmaps and expected heatmaps, respectively. Note that expected heatmaps are normalized to ensure that the maxima on each channel is equal to 1. Following the tutorial reported in DeepHMap, ground truth heatmaps are generated by placing a 2D Gaussian distribution at the ground truth projection for each channel. More details can be seen in Figure 2. In our practice, the immediate output of the last layer of projection grouping module is a feature vector  $V$  with  $N_{out}$  dimension. We flatten the ground truth heatmaps  $O_j$  to construct corresponding probability labels. The standard cross entropy loss for training can be written as

$$L_{pg} = \frac{1}{N_{batch}} \sum_{j=1}^{N_{batch}} \sum_i^{N_{out}} H_j(v_i, s(\hat{v}_i)), \quad (3)$$

where  $H(\cdot)$  represents the binary cross entropy,  $s(\cdot)$  is the softmax function,  $\hat{v}_i$  and  $v_i$  is the  $i$ th element of output vector  $V$  and its corresponding probability label, respectively. With projection grouping module, most unmatched local maxima are removed and projection clusters corresponding to the ground truth are reserved. We test different configuration parameters of projection grouping module (see Figure 2), and the corresponding results can be found in Section 4.



**Figure 2.** Architecture of the projection grouping module. The projection grouping network adopts a residual structure [46] consisting of a fully connected network based feedforward path and a shortcut connection (see case (c) and case (d)). Compared with case (c), case (d) utilizes a fully connected two-layer feedforward network. The projection grouping layer takes merged heatmaps with a shape of  $[w, h, 8]$  as input and generates the same sized heatmaps containing only one peak in each channel. On the input side, merged heatmaps with a shape of  $[w, h, 8]$  are flattened to match the architecture of projection grouping module. On the output side, the immediate output is reshaped to generate filtered projection heatmaps. All layers have rectified linear unit (ReLU) [47] activation function except for the output layer. Dropout layers [48] are employed for the first dense layer, and the softmax function are placed after the add operation. Different configurations of projection grouping layer are detailed in the evaluation section. Additional case (a) and case (b) are plain module without shortcut connection, which are tested for comparison purposes.

### 3.3. Correspondence Learning Based Hypothesis Scoring

Usually, only one peak is reserved for each channel after raw predicted heatmaps travel through the projection grouping module. However, it is still insufficient to directly utilize this maximum to construct correspondence constraints. CNN is inevitably disturbed by some explicit bias from occlusion, background clutter and noise in inference. To minimize these perturbations, we construct a hypothesis pool and throw all 2D–3D correspondences into a correspondence–evaluation network. Correspondence hypotheses are assigned a confidence score and high-confidence correspondences are collected to calculate the object pose. The process of assigning confidence to correspondence hypotheses is referred to as hypothesis scoring.

#### 3.3.1. Generating Hypothesis Pool

We first describe the construction process of the hypothesis pool. As mentioned earlier, only one projection cluster is usually reserved for each channel after projection grouping. Eight

projections of interest centered on the global maxima with a radius of  $R$  are first determined. We use such a projection of interest to accommodate the bias caused by jitters. A total of  $N_{ch}$  correspondence hypotheses are randomly sampled from each projection of interest, including the one corresponding to the peak. Additionally, the sampled points need to have a higher confidence than the predefined threshold. These  $8N_{ch}$  correspondence hypotheses are then fed into the subsequent correspondence–evaluation network.

### 3.3.2. Learning with a Hybrid Loss

The hybrid loss of correspondence learning network [17] consists of a classification term and a regression term. The input correspondences are assigned a weight that indicates whether they are inliers or outliers. Weighted correspondences are then utilized to formulate an essential matrix based regression loss. As shown in Figure 3, the correspondence–evaluation network in our case is formally similar to the correspondence-learning network [17]. The input of our correspondence–evaluation network is 2D–3D correspondence  $c_i = [p_i, P_i]$  instead of keypoint pairs on stereo images. The loss function thus needs to be reformulated to accommodate the new input type. Let  $C = [c_1, c_2, \dots, c_N]$  be a set of 2D–3D correspondences, where  $p_i = [u_i, v_i]$  is the predicted projection in the heatmaps and  $P_i = [x_i, y_i, z_i]$  is the spatial coordinate of BBCs in the object coordinate system. For each object of interest, arbitrary pose can be represented by eight size-specific BBCs. Spatial coordinates of BBCs are thus reused when constructing different correspondences. For each 2D–3D correspondence, the mapping takes the form of a  $3 \times 4$  projection matrix:

$$\forall i, 1 \leq i \leq N, p_i = HP_i. \quad (4)$$

The vector  $p_i$  and  $HP_i$  have the same direction, and Equation (4) thus can be expressed in terms of a vector cross product:

$$p_i \times HP_i = 0. \quad (5)$$

For the over-determined case that has more than six 2D–3D correspondences, the above Equation (5) can be rewritten in the following form:

$$A_i \text{Vec}(H) = 0, \quad (6)$$

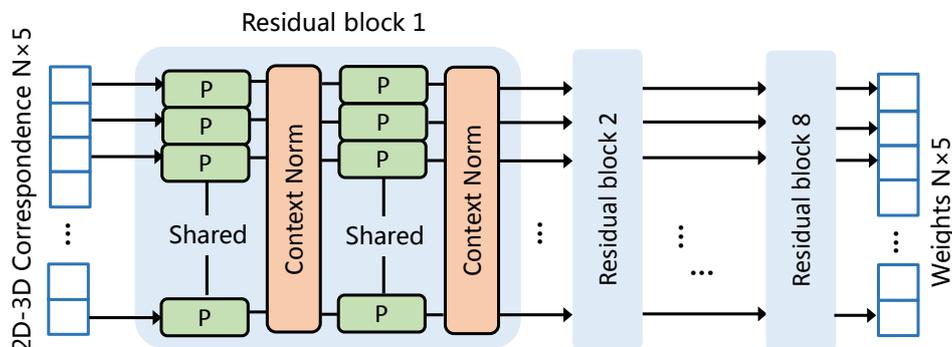
where  $A_i$  denotes the correspondence matrix  $\begin{bmatrix} x_i & y_i & z_i & 1 & 0 & 0 & 0 & 0 & -u_i x_i & -u_i y_i & -u_i z_i \\ 0 & 0 & 0 & 0 & x_i & y_i & z_i & 1 & -v_i x_i & -v_i y_i & -v_i z_i \\ -u_i & & & & & & & & & & \\ -v_i & & & & & & & & & & \end{bmatrix}$ ,  $\text{Vec}(H)$  is the coefficient vector made up of entries from  $H$ . We now construct a  $2N \times 12$  correspondence matrix  $A$  by stacking Equation (6) generated by each correspondence. The projection matrix  $H$  can be computed by performing the singular value decomposition (SVD) of  $A$  and taking the unit singular vector corresponding to the smallest singular value [15].

To suppress possible numerical instability [17,49] in eigendecomposition, the singular value based regression term is replaced with geometry loss:

$$L_{geo} = \frac{1}{N_h} \sum_i^{N_h} \|HP_i - p_i\|_2, \quad (7)$$

where  $N_h$  represents the number of 2D–3D correspondences with a predicted label of 1. The classification term  $L_{cla}$  can be computed by a binary cross-entropy loss, which efficiently rejecting outliers with correspondence classification. Putting both the classification term and geometry term together, the overall loss can be written as:

$$L_{corr\_learn} = \sum_{i=1}^P (\alpha L_{cla} + \beta L_{geo}). \quad (8)$$



**Figure 3.** Architecture of the correspondence–evaluation network. It takes 2D–3D correspondences as input and produces directly correspondence weights. The basic residual block consisting of weight-sharing perceptrons and context normalization. In practice, the multi-layer perceptrons are implemented using *conv1d* in tensorflow [50].

The main differences between the correspondence–evaluation network we use here and the original case [17] are detailed as follows: (1) Instead of 2D–2D correspondences obtained from stereo images, the network here takes 2D–3D correspondences as input and learns the mapping between projection heatmaps and BBCs. (2) Training loss of the network is reformulated by replacing the SVD based regression term to a general reprojection loss (see Equations (7) and (8)). (3) Training dataset of the correspondence evaluation network is different from the conventional 2D case [17] and 3D case [51]. Additional correlation constraints are fused to imitate the projection distribution of 3D BBCs. Only the 10% correspondences with higher confidence are selected to compute the final pose. More training details are given in Section 3.4.

### 3.4. Training Dataset

Our proposed two-stage pipeline can't be trained via an end-to-end fashion because of non-differentiable paths connecting different modules. Three subtasks, prediction of projection heatmaps, projection grouping and correspondence evaluation thus are trained separately. In the first stage, a mixed dataset consisting of synthetic and real samples are generated according to the tutorial described in Ref. [19]. The synthetic samples are collected by accumulating a series of discrete viewpoints, and the real parts are generated by segmenting the masked object of interest and then combining an additional in-plane rotation. This mixed dataset contains 200,000 samples, of which the ratio of synthetic to real is 1 to 1. Hyper parameters of DeepHMap are completely preserved. Note that DeepHMap is object-specific network and we also need to prepare similar object-specific training dataset for each object.

Merged heatmaps are naturally collected to train the projection grouping module. As for the correspondence–evaluation network, it takes a set of 2D–3D correspondences as input. We thus synthesize a series of 2D–3D correspondences by projecting size-specific BBCs to the image coordinate system. Similar to the preparation of training dataset for DeepHMap, a sample set consisting of 200,000 2D–3D correspondences is collected by placing a virtual camera at different viewpoints. Eight BBCs instead of a mesh model are placed at center of the view-sphere. Additional noises and outliers are added to augmented synthesized samples. In practice, the weights in Equation (8) are set to  $\alpha = 1$  and  $\beta = 0.15$ , respectively.

## 4. Evaluations

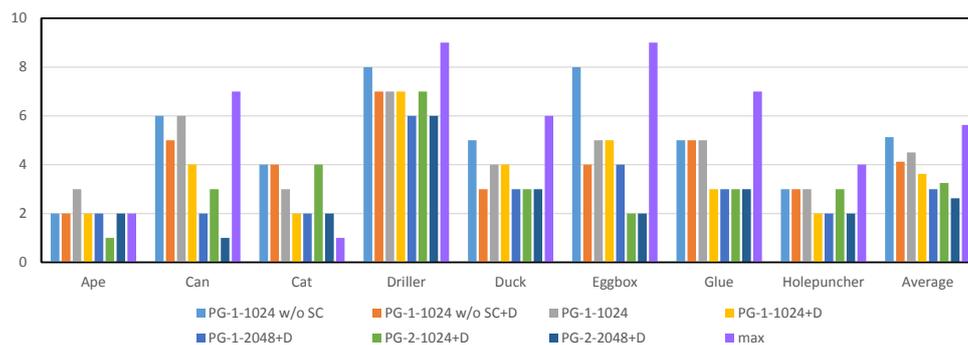
### 4.1. Datasets and Evaluation Metric

In this section, three public datasets: LineMOD dataset [18], Occluded LineMOD dataset [6] and YCB-Video dataset [19] are employed to evaluate the proposed backend and integrated two-stage pipeline. The LineMOD dataset consists of 15 different object sequences and corresponding ground truth pose. The occluded version [6] is generated by selecting images from LineMOD dataset, and these objects occlude each other to a large extent under different viewing directions. The YCB-Video dataset contains 21 different object sequences with significant image noise, illumination changes, background clutter and severe occlusion.

To evaluate the performance of pose estimation algorithms objectively, two popular metrics in this field, 2D reprojection error [34] and ADD | I [18] are employed to define a correctly estimated pose. With the 2D reprojection error, an estimated pose is accepted if the average reprojection error of all model points from the estimated pose and the ground truth pose is below five pixels. ADD depicts a ratio between the average distance and the object's diameter. ADI is specifically designed to deal with symmetrical objects, of which the average distance is computed using the closest point of transformed model points. The default ratio in ADD | I is retained and set to 0.1.

### 4.2. Architecture and Parameter Selection for Projection Grouping Module

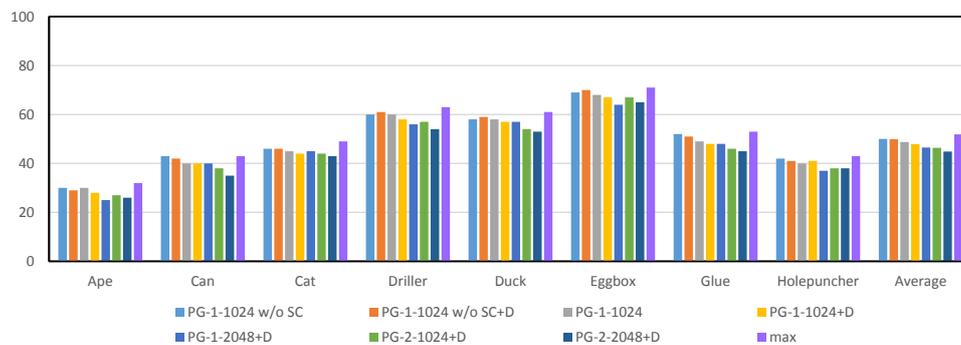
Among the raw merged heatmaps from occluded scenes, multiple local peaks can frequently found in different channels. The one-size-fits-all rule in DeepHMap is not always able to find the optimal projection cluster, which defines a region of interest centered at the ground truth projection with a radius of 10 pixels. To quantify the effect of the projection grouping module, here we count the number of false projection selection (FPS)  $N_{ps}$  per hundred channels. A projection selection is considered correct if its location is inside the corresponding projection cluster. The goal of our projection grouping module is to implicitly learn correlation constraints among projections of BBCs. To best meet the three design principles mentioned above, we test different configurations and report results in Figures 4 and 5. The corresponding results provided by *max* function [11] also have been included. Unless explicitly stated, results from DeepHMap don't utilize feature mapping [52].



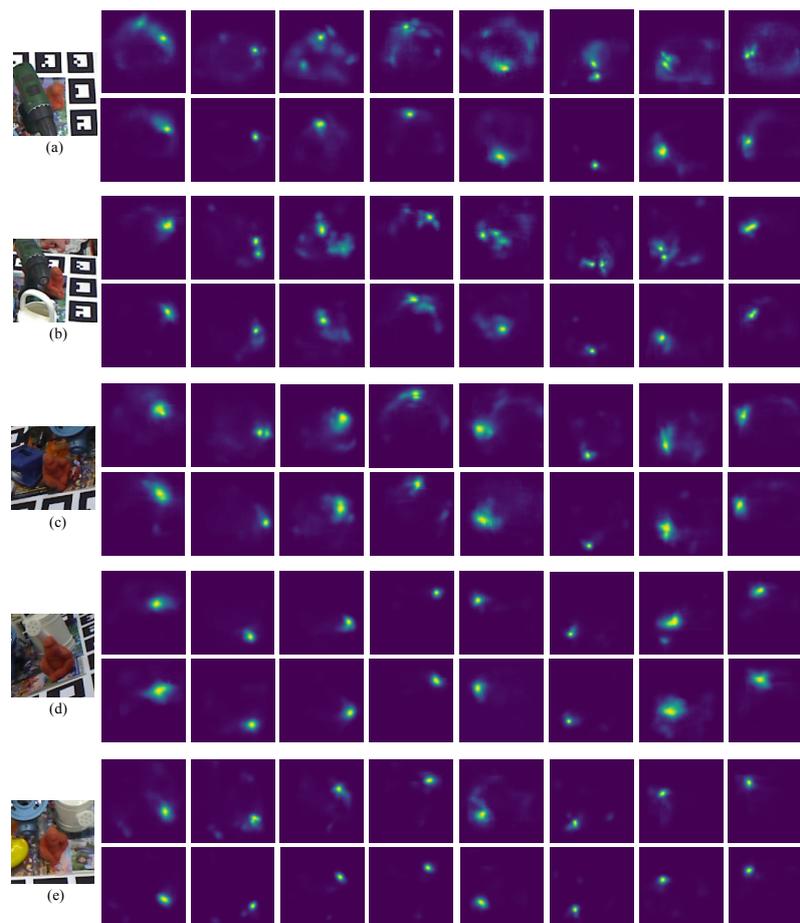
**Figure 4.** Statistics of false projection selection per one hundred channels for different configurations on parts of the LineMOD dataset [18]. A typical configuration can be expressed as PG-x-y w/o SC+D, where PG denotes the abbreviation of projection grouping module, w/o SC indicates that the network doesn't contain shortcut connection (SC), D is the dropout layer,  $x$  and  $y$  represents the number of fully connected layers and dimensionality of the output space, respectively. max refers to the strategy utilized in DeepHMap. For results shown in figure, lower is better.

As shown in Figures 4 and 5, we observe the following results: (1) the projection grouping module with residual architecture and dropout layer achieves best results in FPS metric on both datasets; (2) Compared with the results corresponding to Occluded LineMOD dataset, all test methods give a lower

number of FPS. It is in line with expectations because severe occlusions bring more interference to the inference of network; (3) In addition to the optimal configuration, other configurations of projection grouping module also outperform the *max* function in the baseline [11]. The above experiments consistently demonstrate the effectiveness of projection grouping module. Such improvements can also be seen in Figure 6. For occluded cases, that is (Figure 6a,b) projection heatmaps from DeepHMap++ are much cleaner. As for non-occluded cases, improvements from projection grouping module are visually limited.



**Figure 5.** Statistics of false projection selection per one hundred channels on Occluded LineMOD dataset [6]. All test methods here are the same as Figure 4.

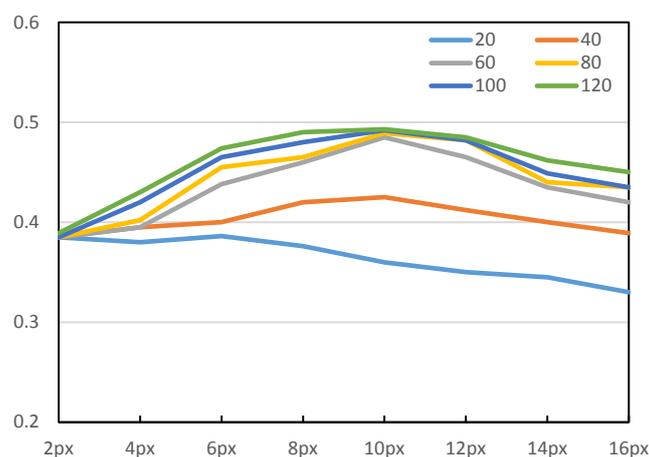


**Figure 6.** Predicted projection heatmaps from different RGB images of Occluded LineMOD dataset [6]. From (a) to (e), the region of interest of different test frames and its corresponding predicted heatmap channels with DeepHMap (up) and projection grouping module (down) are given, respectively.

It benefits by learned correlation constraints and projection grouping module can find the matching projection that doesn't correspond to the global maxima (as shown in Figure 6). For the rest of the evaluation, we use the optimal configuration, that is, PG-2-2048+D.

#### 4.3. Correspondence Evaluation

For the correspondence evaluation module, we evaluate it from three different perspectives. First, we evaluate the performance of correspondence evaluation network with varying sizes of projections of interest and varying numbers of sampled correspondences from each channel (see Figure 7). As mentioned in Sections 3.3 and 3.4, correlation constraints are employed to guide the learning of correspondence evaluation network. To verify the effectiveness of correlation constraints among the training dataset, we test two versions of the correspondence–evaluation networks: trained from dataset with correlation constraints (CorrNet) and without correlation constraints (CorrNet w/o CC). Here, correlation constraints are evaluated as the second factor. For the non-constraint case, we follow the similar procedure of a view-sphere based method [18]. For each viewpoint, we randomize the 2D position of projections and achieve the corresponding 3D reference points by a back-projection function. Third, we evaluate the correspondence–evaluation module against a RANSAC based strategy employed in DeepHMap.



**Figure 7.** Evaluation of our proposal with a varying radius of projection of interest (POI) and different sampled correspondences. A horizontal axis denotes the radius of POI in pixels. The vertical axis denotes the fraction of correctly estimated scenes under the 2D reprojection error metric.

Regarding the radius of POI, Figure 7 shows that the increase of precision tends to saturation at 10  $px$ . As for the number of sampled correspondences, we can find that increasing from 60 to 80 or more only slightly affects the performance of the correspondence evaluation network. For the rest of the evaluation, we thus use the parametric values of  $r = 10 px, n = 60$ .

With utilizing the identified radius of POI and the number of sampled correspondences, we then begin to evaluate the impact of correlation constraints on the CorrNet. It should be noted that the evaluation of DeepHMap on the LineMOD dataset hasn't been given, and thus we list the corresponding results from BB8 [8] as a substitute. Table 1 shows that both of the two versions significantly outperform BB8 [11] on the LineMOD dataset [18]. This is mainly because of the specially designed network for weighted correspondence and projection grouping, and boosting from DeepHMap. In the case of without correlation constraints, the average accuracy of CorrNet w/o CC is about 9.7% higher than BB8 [11] in ADD | I metric, and about 5.3% higher in the 2D reprojection error metric. In addition, correlation constraints among the training dataset can further improve the average accuracy of CorrNet w/o CC by 1.8% under ADD | I metric, and about 0.7% in a

2D-reprojection error metric. It proves the validity of correlation constraints on the correspondence selection.

Similar test results from Occluded LineMOD dataset [6] can be found in Table 2. Margins between RANSAC based strategy in DeepHMap and two versions of CorrNet reach an average of 1.6% and 1.0% in ADD | I metric, 5.2% and 3.0% in the 2D reprojection error metric. It confirms once again that correlation constraints can further improve the performance of CorrNet. CorrNet has a stronger ability to handle fake correspondences than the RANSAC based strategy [11].

**Table 1.** Pose estimation results of two different versions: CorrNet and CorrNet w/o CC on the LineMOD dataset [18]. The best results for each term are shown in bold.

Sequence	ADD   I			2D Reprojection Error		
	BB8 [8]	CorrNet w/o CC	CorrNet	BB8 [8]	CorrNet w/o CC	CorrNet
Ape	40.4	49.7	<b>51.2</b>	96.6	97.7	<b>98.2</b>
Benchvise	91.8	93.0	<b>93.5</b>	90.1	97.5	<b>98.1</b>
Camera	55.7	61.7	<b>62.9</b>	86.0	96.2	<b>96.5</b>
Can	64.1	71.5	<b>72.9</b>	91.2	98.1	<b>98.4</b>
Cat	62.6	65.5	<b>67.1</b>	<b>98.8</b>	98.7	<b>98.8</b>
Driller	74.4	81.5	<b>82.6</b>	80.9	88.8	<b>90.1</b>
Duck	44.3	55.7	<b>59.0</b>	92.2	97.0	<b>97.3</b>
Eggbox	57.8	74.6	<b>77.0</b>	91.0	95.2	<b>95.5</b>
Glue	41.2	83.4	<b>85.1</b>	92.3	97.8	<b>98.5</b>
Holepuncher	67.2	73.9	<b>75.7</b>	95.3	97.8	<b>98.2</b>
Iron	84.7	87.8	<b>89.3</b>	84.8	88.5	<b>89.9</b>
Lamp	76.5	80.7	<b>83.1</b>	75.8	86.5	<b>88.1</b>
Phone	54.0	62.1	<b>65.6</b>	85.3	90.6	<b>91.0</b>
Average	62.7	72.4	<b>74.2</b>	89.3	94.6	<b>95.3</b>

**Table 2.** Pose estimation results of CorrNet (ours) and RANSAC [11] based method on an Occluded LineMOD dataset [6]. The best results for each term are shown in bold.

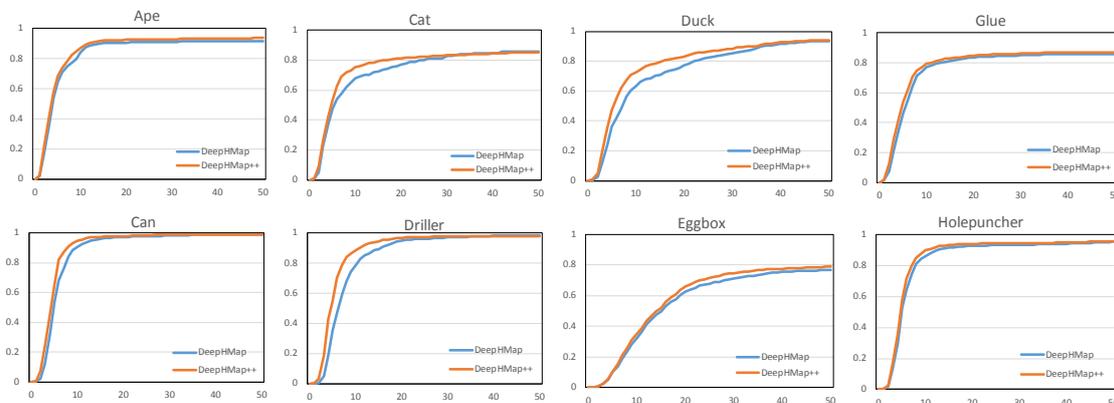
Sequence	ADD   I			2D Reprojection Error		
	RANSAC [11]	CorrNet w/o CC	CorrNet	RANSAC [11]	CorrNet w/o CC	CorrNet
Ape	16.5	17.0	<b>17.3</b>	64.7	66.5	<b>68.6</b>
Can	42.5	45.8	<b>49.2</b>	53.0	61.7	<b>64.9</b>
Cat	2.8	2.9	<b>3.0</b>	47.9	51.4	<b>53.3</b>
Driller	47.1	54.6	<b>57.7</b>	35.1	48.5	<b>55.0</b>
Duck	11.0	12.1	<b>13.2</b>	36.1	39.5	<b>47.3</b>
Eggbox	24.7	24.9	<b>25.0</b>	10.3	10.3	<b>10.4</b>
Glue	39.5	39.7	<b>39.9</b>	44.9	51.7	<b>53.4</b>
Holepuncher	<b>21.9</b>	<b>21.9</b>	<b>21.9</b>	52.9	56.6	<b>57.6</b>
Average	25.8	27.4	<b>28.4</b>	43.1	48.3	<b>51.3</b>

#### 4.4. Results from the Full Pipeline

We now evaluate our full pipeline on two datasets with severe occlusion, namely Occluded LineMOD dataset [6], and YCB-Video dataset [19]. For comparison purposes, we have employed two state-of-the-art methods, that is, PoseCNN [19] and DeepHMap [11]. Note that all methods in the evaluation section take only RGB images as input. Especially for the YCB-Video dataset, the area under the accuracy-threshold curve (AUC) [19] is utilized as an additional metric.

As depicted in Figure 8, a more complete comparison between DeepHMap and DeepHMap++ is given. For all eight sequences from the Occluded LineMOD dataset [6], DeepHMap++ steadily achieves better results than DeepHMap under different pixel thresholds. With the 2D reprojection error metric, a smaller pixel threshold means more accurate estimation. It is not unusual to find that the boosting of DeepHMap++ is more obvious under a low-threshold phase that ranges from 0 px to

30 *px*. This is because, when a test scene corresponds to a larger pixel threshold, it means that the estimated pose deviates significantly from the ground truth. Dealing with such challenging scenes is very difficult for both projection grouping module and correspondence evaluation module. Thus, the improvement of DeepHMap++ becomes limited when pixel threshold reaches a high level that is bigger than 30 *px*.



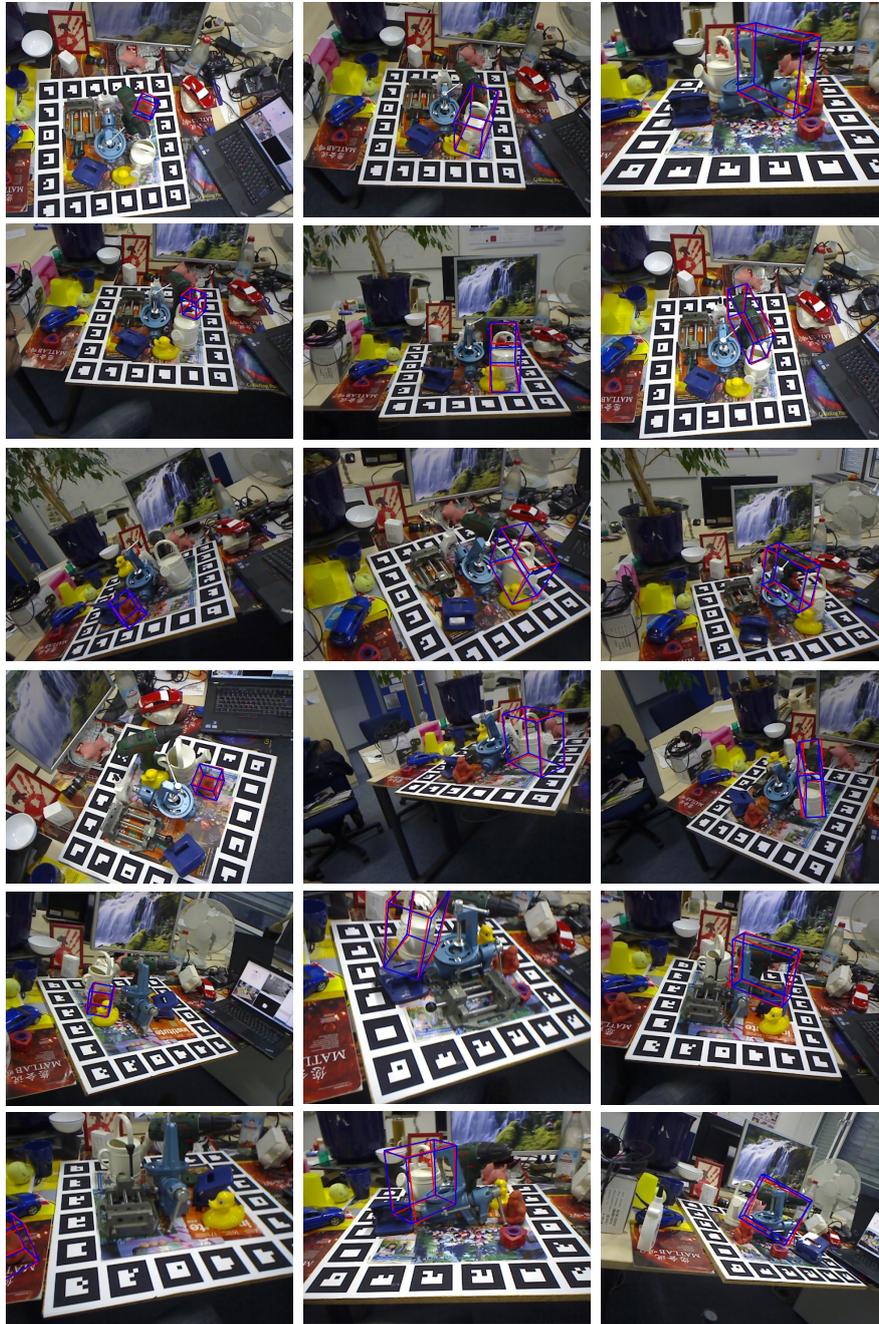
**Figure 8.** Evaluations on an Occluded LineMOD dataset [6]. The curve represents accuracy vs. pixel threshold in a 2D reprojection error metric. The vertical axis denotes a fraction of correctly estimated scenes. The horizontal axis denotes pixel threshold.

**Table 3.** Comparisons with state-of-the-art methods on YCB-Video dataset [19]. We report the AUC scores, ADD||I and 2D reprojection error for the 21 image sequences of YCB-Video dataset. The best results for each term are shown in bold.

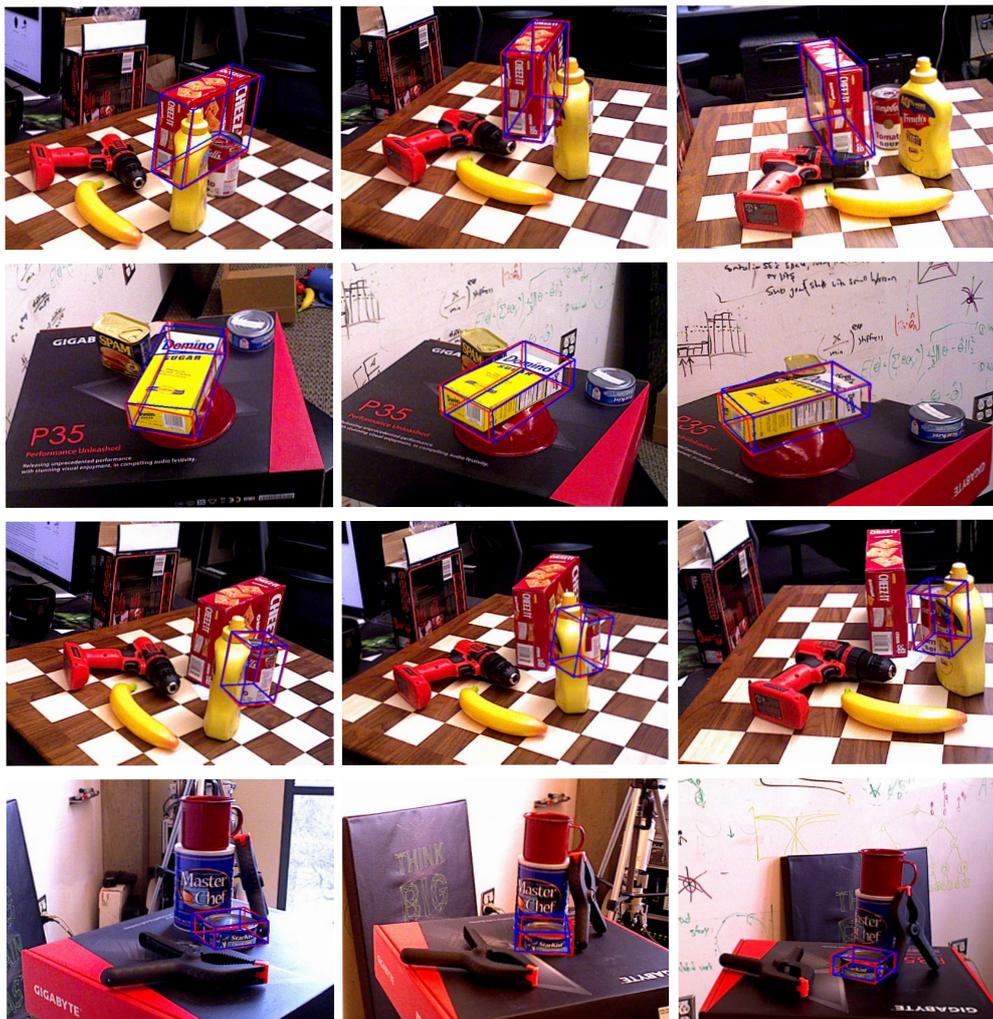
Sequence	PoseCNN [19]			DeepHMap [11]			DeepHMap++		
	AUC	ADD  I	2D Repr.	AUC	ADD  I	2D Repr.	AUC	ADD  I	2D Repr.
002 master chef can	50.1	3.6	0.1	68.5	32.9	9.9	<b>75.8</b>	<b>40.1</b>	<b>20.1</b>
003 cracker box	52.9	25.1	0.1	74.7	62.6	24.5	<b>78.0</b>	<b>69.5</b>	<b>34.5</b>
004 sugar box	68.3	40.3	7.1	74.9	44.5	47.0	<b>76.5</b>	<b>49.7</b>	<b>58.9</b>
005 tomato soup can	66.1	25.5	5.2	68.7	31.1	41.5	<b>72.1</b>	<b>36.1</b>	<b>49.8</b>
006 mustard bottle	<b>80.8</b>	<b>61.9</b>	6.4	72.6	42.0	42.3	78.9	57.9	<b>60.1</b>
007 tuna fish can	<b>70.6</b>	<b>11.4</b>	3.0	38.2	6.8	7.1	51.6	9.8	<b>19.5</b>
008 pudding box	62.2	14.5	5.1	82.9	58.4	43.9	<b>85.6</b>	<b>67.2</b>	<b>56.8</b>
009 gelatin box	74.8	12.1	15.8	82.8	42.5	62.1	<b>86.7</b>	<b>59.1</b>	<b>76.8</b>
010 potted meat can	59.5	18.9	23.1	66.8	37.6	38.5	<b>70.1</b>	<b>42.0</b>	<b>42.3</b>
011 banana	<b>72.1</b>	<b>30.3</b>	0.3	44.9	16.8	8.2	47.9	19.3	<b>10.5</b>
019 pitcher base	53.1	15.6	0.0	70.3	57.2	15.9	<b>71.8</b>	<b>58.5</b>	<b>19.8</b>
021 bleach cleanser	50.2	21.2	1.2	67.1	65.3	12.1	<b>69.1</b>	<b>69.4</b>	<b>18.5</b>
024 bowl	<b>69.8</b>	12.1	4.4	58.6	25.6	16.0	60.2	<b>27.7</b>	<b>18.1</b>
025 mug	<b>58.4</b>	5.2	0.8	38.0	11.6	20.3	43.4	<b>12.9</b>	<b>26.3</b>
035 power drill	55.2	29.9	3.3	72.6	46.1	40.9	<b>76.8</b>	<b>51.8</b>	<b>50.1</b>
036 wood block	<b>61.8</b>	10.7	0.0	57.7	34.3	2.5	61.3	<b>35.7</b>	<b>2.8</b>
037 scissors	35.3	<b>2.2</b>	0.0	30.9	0.0	0.0	<b>42.9</b>	2.1	<b>6.7</b>
040 large marker	<b>58.1</b>	3.4	<b>1.4</b>	46.2	3.2	0.0	47.6	<b>3.6</b>	0.8
051 large clamp	<b>50.1</b>	<b>28.5</b>	0.3	42.4	10.8	0.0	44.1	11.2	<b>8.7</b>
052 extra large clamp	46.5	19.6	0.6	48.1	29.6	0.0	<b>51.9</b>	<b>30.9</b>	<b>0.8</b>
061 foam brick	<b>85.9</b>	54.5	0.0	82.7	51.7	52.4	84.1	<b>55.4</b>	<b>59.7</b>
Average	61.0	21.3	3.7	61.4	33.8	23.1	<b>65.5</b>	<b>38.6</b>	<b>30.6</b>

Comparisons between DeepHMap++, DeepHMap and PoseCNN are listed in Table 3. For all object sequences from YCB-Video dataset, DeepHMap++ consistently improves DeepHMap, which utilizes RANSAC based correspondence sampling and *max* function based projection grouping in three different metrics. For another baseline [19], semantic labeling and object center are jointly employed intermediate cues. However, the entire image is directly cast into CNN for building the mapping from image space to object center. This holistic scheme is more sensitive to foreground occlusion than

both DeepHMap and DeepHMap++ using local input. Local feature input plus a specially designed back-end ensures that DeepHMap++ achieves best results over most of the entries. We also show some qualitative results on both datasets in Figures 9 and 10, respectively.



**Figure 9.** Estimated 6D pose on an Occluded LineMOD dataset [6]. The red and blue bounding boxes denote the ground truth and results estimated by DeepHMap++, respectively. The left column is the results of Ape sequence. The middle column is the results from Can sequence. The right column is the results from Driller sequence.



**Figure 10.** Estimated 6D pose on YCB-Video dataset [19]. The ground truth is shown in red, and estimated results with DeepHMap++ are shown in blue. The four rows (from up to down) correspond to test images from 003\_cracker\_box sequence, 004\_sugar\_box sequence, 005\_tomato\_soup\_can sequence and 007\_tuna\_fish\_can sequence, respectively.

#### 4.5. Runtime Analysis

Our current implementation is written in python on an Ubuntu machine with an intel E5-2640 CPU (intel, Santa Clara, CA, USA) and NVIDIA Geforce GTX1080Ti GPU (NVIDIA, Santa Clara, CA, USA). The network part is built on a large-scale machine learning library-tensorflow [50]. In the first stage, the projection heatmap prediction takes 80 ms for 64 patches. Parallel processing can significantly reduce the prediction time to 20 ms. Benefiting from the simple architecture of projection grouping module, it only takes 3 ms to complete this subtask. Subsequent correspondence evaluation network takes 20 ms to assign correspondence weights and compute the full DoF pose parameters. For a  $640 \text{ px} \times 480 \text{ px}$  image, it takes about 130 ms for pose detection via a sliding window fashion and goes down to 60 ms with a parallel trick. The main runtime statistics are listed in Table 4.

**Table 4.** The runtime statistics of different subparts in the complete pipeline.

	Processing Step	Time
First stage	Projection heatmap predictions	80 ms
	Projection heatmap predictions (in parallel)	20 ms
Second stage	Projection grouping	3 ms
	Correspondence evaluation	20 ms
Full pipeline		$80 + 3 + 20 = 103$ ms
Full pipeline (in parallel)		$20 + 3 + 20 = 43$ ms

## 5. Conclusions and Future Work

We have improved the back-end of a two-stage pipeline to recover the 6D pose of rigid objects under challenging scenes. With a simple fully connected module, the projection ambiguity can be better addressed than the one-size-fits-all strategy in DeepHMaps. The proposed projection grouping module learns correlation constraints of different BBCs and reduces the number of false projection selections. A corresponding-evaluation network is then employed to achieve weighted correspondences, as opposed to RANSAC based strategy. The above mentioned efforts have enabled the proposed method to outperform state-of-the-art solutions on three public benchmarks. Meanwhile, these refinements don't introduce too much computing burden, which indicates the great potential of our method in real-time applications.

In the future, an interesting direction is to add a branch on the backbone for object segmentation. This branch can provide additional regularization to some extent. Another line is to fuse the improved two-stage approach into a pose tracking framework. In a standard pipeline of pose tracking, pose parameters from the previous frame can be reused to replace the pose detection step in DeepHMap. In addition, tuning the architecture of a network to achieve an end-to-end training is beneficial to final results.

**Author Contributions:** M.F. conceived of and designed the experiments. M.F. performed the experiments. M.F. analyzed the data. W.Z. supervised this work.

**Funding:** This research was funded by the National Science Foundation of China Grant No. 51505470.

**Acknowledgments:** The authors would like to thank Stefan Hinterstoisser (Google), Eric Brachmann (Heidelberg University) and Xiang Yu (NVIDIA) for making related datasets freely available to the public.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Lepetit, V.; Fua, P. *Monocular Model-Based 3D Tracking of Rigid Objects*; Now Publishers Inc.: Hanover, MA, USA, 2005; pp. 1–89.
- Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [[CrossRef](#)]
- Rothganger, F.; Lazebnik, S.; Schmid, C.; Ponce, J. 3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *Int. J. Comput. Vis.* **2006**, *66*, 231–259. [[CrossRef](#)]
- Crivellaro, A.; Rad, M.; Verdie, Y.; Yi, K. M.; Fua, P.; Lepetit, V. Robust 3D object tracking from monocular images using stable parts. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1465–1479. [[CrossRef](#)] [[PubMed](#)]
- Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [[CrossRef](#)]
- Brachmann, E.; Krull, A.; Michel, F.; Gumhold, S.; Shotton, J.; Rother, C. Learning 6D Object Pose Estimation using 3D Object Coordinates. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 536–551.
- 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 Proceedings of the International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.

8. Nigam, A.; Penate-Sanchez, A.; Agapito, L. Detect Globally, Label Locally: Learning Accurate 6-DOF Object Pose Estimation by Joint Segmentation and Coordinate Regression. *IEEE Robot. Autom. Lett.* **2018**, *3*, 3960–3967. [[CrossRef](#)]
9. 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 Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Santa Rosa, CA, USA, 24–31 March 2017; pp. 880–888.
10. 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, Santiago, Chile, 7–13 December 2015; pp. 4391–4399.
11. 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, Munich, Germany, 8–14 September 2018.
12. Brachmann, E.; Rother, C. Learning less is more-6d camera localization via 3D surface regression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.
13. Krull, A.; Brachmann, E.; Nowozin, S.; Michel, F.; Shotton, J.; Rother, C. Poseagent: Budget-constrained 6D object pose estimation via reinforcement learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
14. Tekin, B.; Sinha, S. N.; Fua, P. Real-time seamless single shot 6D object pose prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.
15. Hartley, R.I.; Zisserman, A. *Multiple View Geometry in Computer Vision*; 2nd ed.; Cambridge University Press: Cambridge, UK, 2000.
16. Brachmann, E.; Krull, A.; Nowozin, S.; Shotton, J.; Michel, F.; Gumhold, S.; Rother, C. DSAC—Differentiable RANSAC for Camera Localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
17. Yi, K.M.; Trulls, E.; Ono, Y.; Lepetit, V.; Salzmann, M.; Fua, P. Learning to Find Good Correspondences. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.
18. 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 Proceedings of the Asian Conference on Computer Vision, Daejeon, Korea, 5–9 November 2012; pp. 548–562.
19. Xiang, Y.; Schmidt, T.; Narayanan, V.; Fox, D. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. In Proceedings of the Robotics: Science and Systems, Pittsburgh, PA, USA, 26–30 June 2018.
20. Tejani, A.; Kouskouridas, R.; Doumanoglou, A.; Tang, D.; Kim, T.K. Latent-Class Hough Forests for 6 DoF Object Pose Estimation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 119–132. [[CrossRef](#)] [[PubMed](#)]
21. 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, Boston, MA, USA, 7–12 June 2015.
22. Balntas, V.; Doumanoglou, A.; Sahin, C.; Sock, J.; Kouskouridas, R.; Kim, T.K. Pose Guided RGBD Feature Learning for 3D Object Pose Estimation. In Proceedings of the International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3856–3864.
23. 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, Las Vegas, NE, USA, 27–30 June 2016.
24. Li, C.; Bai, J.; Hager, G.D. A Unified Framework for Multi-View Multi-Class Object Pose Estimation. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
25. Mitash, C.; Boularias, A.; Bekris, K. Physics-based Scene-level Reasoning for Object Pose Estimation in Clutter. *arXiv* **2018**, arXiv:1806.10457.
26. Wu, J.; Zhou, B.; Russell, R.; Kee, V.; Wagner, S.; Hebert, M.; Torralba, A.; Johnson, D. Real-Time Object Pose Estimation with Pose Interpreter Networks. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Madrid, Spain, 1–5 October 2018.
27. Do, T.T.; Cai, M.; Pham, T.; Reid, I. Deep-6DPose: Recovering 6D Object Pose from a Single RGB Image. *arXiv* **2018**, arXiv:1802.10367.

28. Periyasamy, A.S.; Schwarz, M.; Behnke, S. Robust 6D Object Pose Estimation in Cluttered Scenes using Semantic Segmentation and Pose Regression Networks. *arXiv* **2018**, arXiv:1810.03410.
29. 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, Munich, Germany, 8–14 September 2018; pp. 712–729.
30. Doumanoglou, A.; Balntas, V.; Kouskouridas, R.; Kim, T.K. Siamese regression networks with efficient mid-level feature extraction for 3d object pose estimation. *arXiv* **2016**, arXiv:1607.02257.
31. 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 Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 205–220.
32. Zhang, H.; Cao, Q. Combined Holistic and Local Patches for Recovering 6D Object Pose. In Proceedings of the International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2219–2227.
33. Zeng, A.; Yu, K.T.; Song, S.; Suo, D.; Walker, E.; Rodriguez, A.; Xiao, J. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In Proceedings of the IEEE International Conference on Robotics and Automation, Singapore, 29 May–3 June 2017.
34. Brachmann, E.; Michel, F.; Krull, A.; Ying Yang, M.; Gumhold, S. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
35. Jafari, O.H.; Mustikovela, S.K.; Pertsch, K.; Brachmann, E.; Rother, C. iPose: Instance-Aware 6D Pose Estimation of Partly Occluded Objects. In Proceedings of the Asian Conference on Computer Vision, Perth, Australia, 4–6 December 2018.
36. Kehl, W.; Manhardt, F.; Tombari, F.; Illic, 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, Venice, Italy, 22–29 October 2017.
37. Pavlakos, G.; Zhou, X.; Chan, A.; Derpanis, K.G.; Daniilidis, K. 6-dof object pose from semantic keypoints. In Proceedings of the IEEE International Conference on Robotics and Automation, Singapore, 29 May–3 June 2017.
38. Michel, F.; Kirillov, A.; Brachmann, E.; Krull, A.; Gumhold, S.; Savchynskyy, B.; Rother, C. Global hypothesis generation for 6D object pose estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
39. Sock, J.; Kim, K.I.; Sahin, C.; Kim, T.K. Multi-Task Deep Networks for Depth-Based 6D Object Pose and Joint Registration in Crowd Scenarios. *arXiv* **2018**, arXiv:1806.03891.
40. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
41. Krull, A.; Brachmann, E.; Michel, F.; Yang, M.Y.; Gumhold, S.; Rother, C. Learning Analysis-by-Synthesis for 6D Pose Estimation in RGB-D Images. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015.
42. Sahin, C.; Kim, T.K. Recovering 6D Object Pose: A Review and Multi-modal Analysis. In Proceedings of the European Conference on Computer Vision Workshops on Assistive Computer Vision and Robotics, Munich, Germany, 8–14 September 2018.
43. Opitz, M.; Waltner, G.; Poier, G.; Possegger, H.; Bischof, H. Grid loss: Detecting occluded faces. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 386–402.
44. Yang, S.; Luo, P.; Loy, C.C.; Tang, X. Faceness-Net: Face Detection through Deep Facial Part Responses. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1845–1859. [[CrossRef](#)] [[PubMed](#)]
45. Jaderberg, M.; Simonyan, K.; Zisserman, A.; Kavukcuoglu, K. Spatial transformer networks. In Proceedings of the Advances In Neural Information Processing Systems, Montreal, QC, Canada, 11–12 December 2015; pp. 2017–2025.
46. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 27–30 June 2016.
47. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 21–24 June 2010; pp. 807–814.

48. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
49. Dang, Z.; Yi, K.M.; Hu, Y.; Wang, F.; Fua, P.; Salzmann, M. Eigendecomposition-free Training of Deep Networks with Zero Eigenvalue-based Losses. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
50. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A System for Large-Scale Machine Learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, Savannah, GA, USA, 2–4 November 2016.
51. Ferraz, L.; Binefa, X.; Moreno-Noguer, F. Very Fast Solution to the PnP Problem with Algebraic Outlier Rejection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014.
52. 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, Salt Lake City, UT, USA, 18–22 June 2018.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).