3D Key-points Estimation From Single-view RGB Images

Mohammad Zohaib[1,2][0000−0003−2259−4121], Matteo Taiana[1][0000−0003−1759−2447], Milind Gajanan Padalkar[1][0000−0002−0342−5448], and Alessio Del Bue[1][0000−0002−2262−4872]

[1] Pattern Analysis & Computer Vision (PAVIS), Italian Institute of Technology, Genoa, Italy
[2] Department of Marine, Electrical, Electronic and Telecommunications Engineering, University of Genoa, Italy
{mohammad.zohaib, matteo.taiana, milind.padalkar, alessio.delbue}@iit.it

Abstract. The paper presents an end-to-end approach that leverages images for estimating an ordered list of 3D key-points. Most of the existing methods either use point clouds or multiple RGB/depth images to estimate 3D key-points, whereas the proposed approach requires only a single-view RGB image. It is based on three steps: extracting latent codes, computing pixel-wise features, and estimating 3D key-points. It also computes a confidence score of every key-point that enables it to predict a different number of key-points based on an object's shape. Therefore, unlike existing approaches, the network can be trained to address several categories at once. For evaluation, we first estimate 3D key-points for two views of an object and then use them for finding a relative pose between the views. The results show that the average angular distance error of our approach (6.39°) is 8.01° lower than that of KP-Net (14.40°) [1].

Keywords: 3D Key-points · Single-view RGB images · Pose estimation

1 Introduction

Finding objects' position and pose in images is a necessary step for solving higher level tasks such as object search, manipulation, navigation in cluttered environments, path planning, and human-robot interaction. Recent research reveals that estimating object location and pose can be improved significantly with the help of 3D key-points. It is due to the fact that they provide information about Points of Interest (PoI) and are invariant to transformation i.e., rotations, scale, etc. [1–6]. Moreover, they also contain semantic information, which is helpful while reasoning on correspondences between points in two shapes [3, 7, 8].

Most of the recent studies use 3D key-points for various human related applications including joint detection, motion capturing, pose estimation, etc., which deal with a single category (human) and a fixed number of key-points [9–17]. On the other hand, key-points are also used in applications related to rigid objects (i.e. car, chair, etc.), where an object's structure and the number of

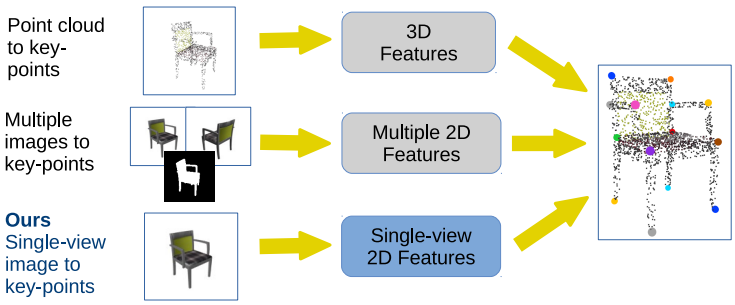


Fig. 1: Comparison with the other paradigms. Some existing methods use point clouds (top) or multiple images representing different views of an object (middle) as inputs and compute 2D/3D features for key-points estimation. In comparison, the proposed approach considers a single-view RGB image, extracts object 2D features, and use them for estimating 3D key-points (bottom).

key-points may vary depending on the category. To simplify the problem, the existing approaches train their network separately for every category for a fixed number of key-points. In the literature, most of the works including [7, 18–20] compute 3D key-points from 3D point clouds. On the other hand, a method proposed in [1] uses RGB images. However, it estimates 3D key-points in the form of 2D pixels and associated depth values. In comparison, we present an approach that uses a single-view RGB image and estimates an ordered list of key-points in 3D space. An overview of our approach is shown in Fig. 1, highlighting the key difference with the existing approaches. Our main contributions are as follows;

- The proposed approach estimates key-points from a single-view RGB image.
- Unlike the existing approaches, the proposed approach estimates the confidence score for every key-point that allows it to predict the different number of key-points based on the object's shape.
- The estimated key-points provide order-wise semantic information that is independent of the object's view.

The remainder of the paper is organized as follows: Section 2 presents a literature review, Section 3 describes the proposed system, and Section 4 discusses the experiments we performed and compares the results with state-of-the-art (SOTA) techniques. Finally, conclusions are summarised in Section 5.

2 Related work

Key-points provide an object's structural information, which can be utilized in geometric reasoning. The popularity stems from the fact that they require minimum processing resources and are easy to handle in comparison to complete 3D point clouds of an object or 2D pixels of an image. Moreover, in some cases, they also contain semantic information by ensuring their unique order.

In literature, most of the approaches use point clouds for estimating 3D key-points. Adamczyk et al. present an evolutionary algorithm for the selection of visual key-points from unordered sets to address the classification problem [21]. An approach that estimates category-specific 3D key-points in an unsupervised way is presented in [22]. It considers linear symmetric shapes and produces order-wise correspondences with consistent semantics. Jakab et al. [23] use key-points for aligning two shapes. Their network takes two shapes and finds the key-points for shape deformation from a set of randomly sampled surface points. Chen et al. [24] present an unsupervised approach that computes key-points from the object's point cloud to represent good abstraction and approximation of the input 3D shape. This approach encodes local features using PointNet++ and uses them for producing a set of ordered 3D key-points. You et al. [27] present a method that uses geodesic consistency loss for producing dense semantic embeddings. Sun et al. present an unsupervised approach for object parts decomposition using key-points estimation [31]. Their network is trained to compute K semantic correspondence key-points by feeding two randomly rotated versions of an object in the form of point clouds. The SK-Net proposed in [25] generates random spatial key-points in 3D space and converges them to an object's point cloud by learning geometric features. Unlike the other approaches, the spatial key-points are not a part of the object's point cloud.

Other works have focused on depth images and/or RGB. Georgakis et al. present an approach that uses RGB and depth images to compute object 3D pose by matching predicted key-points to the corresponding CAD model [26]. He et al. present a point-wise 3D key-points voting network that uses key-points for calculating object pose in six Degrees of Freedom (6DoF) [28]. They train the network using RGBD images and predict key-points used by the least-squares fitting method for estimating the object's 6D pose. Another RGBD images based approach is presented in [29] that uses estimated 3D key-points for tracking an object's pose. Although their network does not require 3D shapes during training, they are however, required test objects that are relatively similar to those used as training samples [30]. Barabanau et al. [32] present an approach that estimates 2D key-points from an RGB image and transforms these key-points to a 3D model of an object taken from a predefined set of 5 CAD templates only. The intrinsic camera parameters are known. Lu et al. present an approach that uses key-points for finding the pose of a robotic arm [33]. Initially, key-points are sampled on the kinematic chain and are filtered in order to select optimal ones using RANSAC [34]. A similar approach that finds semantic correspondence between two images using both appearance and geometry reasoning by incorporating 2D key-points is presented by Han et al. in [35]. The approach uses these semantic correspondences for producing a warped version of two images. Suwajanakorn et al. compute 3D key-points in an unsupervised way by using two views of an object in different poses and knowledge of the object category [1]. During inference, they use single-view RGB images. However, their estimates are in the form of 2D pixels and depths. In comparison, we present a supervised approach to estimate key-points in 3D space from a single-view RGB image.

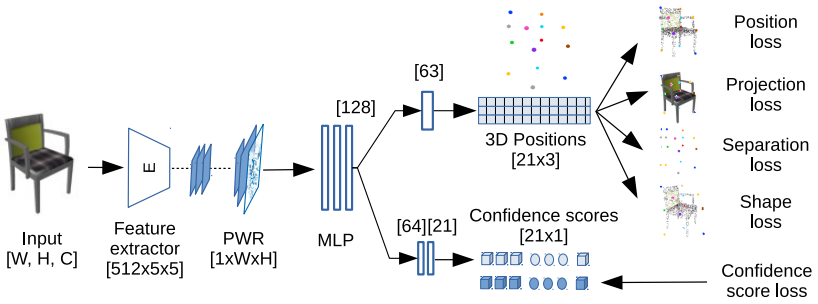


Fig. 2: The proposed architecture. An RGB image is fed to a feature extractor to produce object features that are up-sampled in order to achieve a Pixel-wise Representation (PWR). Finally, a Multilayer Perceptron (MLP) is added that uses PWR for estimating 21 key-points in 3D space along with confidence scores.

3 Methodology

Given an RGB image, our work aims at estimating an ordered list of 3D key-points that are semantically and geometrically consistent across different instances of an object category. For this, an end-to-end approach is proposed that extracts an object's features from an image, computes Pixel-wise Representation (PWR) and uses the representation for estimation of 3D key-points along with confidence scores. The architecture of the approach is illustrated in Fig. 2.

The presented approach is based on three modules. The first module (feature extractor) takes an RGB image as input and produces feature vectors. These extracted features are converted to PWR in the second module. The PWR has the same width and height as the input image. However, instead of representing the RGB value, every pixel represents a feature for the corresponding pixel of the input image. The third module contains a Multilayer Perceptron (MLP) based on four linear layers. The PWR features are flattened to a 1D tensor before feeding to the MLP. The MLP uses them for estimating the 21 key-points. For every key-point, a position in 3D space $[x, y, z]$ and a confidence score (from 0 to 1) is computed. The confidence score reflects how confident the network is that the key-point exists for the object. If such a value is greater than 0.5, it means that the predicted key-point exists for the object, and it is considered as a valid key-point. Otherwise, it is discarded. In this way, the network separates an object's valid key-points from the predicted 21 key-points. So, the total number of valid key-points could be different for different shapes of objects.

3.1 Training loss

The network is trained separately for every category (as followed in literature) as well as jointly for all the categories. We found that the results with both methods are approx. the same. So, we report the results of a network trained jointly for all

the categories. The network minimizes five losses: 3D position loss, 2D projection loss, separation loss, shape consistency loss and the confidence score loss. The **3D position loss** ($\mathcal{L}_{pos}$) measures how accurate the 3D position corresponding to the predicted key-point is, w.r.t. the ground truth. For this we compute Mean Square Error (MSE) between 3D positions of predicted $\mathcal{P} = \{p_i | i = 1, ..., N_p\}$ and ground truth $\mathcal{Q} = \{q_i | i = 1, ..., N_q\}$ key-points as:

$$\mathcal{L}_{pos} = \frac{1}{N_q} \sum_{i=1}^{N_q} \left\| p_i - q_i \right\|_2^2, \tag{1}$$

where N_q is the total number of ground truth key-points, which could be up to 21. We skip the extra predicted key-points that are not valid for an object (w.r.t. ground truth). In order to predict a more accurate position of the key-points, we also compute loss in 2D space. To do so, both the valid estimated and ground truth key-points are transformed from 3D to 2D pixel coordinates using the known transformation T (camera intrinsic and extrinsic) [1]. The **2D projection loss** ($\mathcal{L}_{proj}$) is computed by taking the Mean Absolute Error (MAE) between estimated and ground truth 2D pixels as:

$$\begin{aligned} p_i = [x_{p_i}, y_{p_i}, z_{p_i}]^\top, \;\; q_i = [\bar{x}_{q_i}, \bar{y}_{q_i}, \bar{z}_{q_i}]^\top \\ [u_i, v_i]^\top = T(p_i), \;\; [\bar{u}_i, \bar{v}_i]^\top = T(q_i) \\ \mathcal{L}_{Proj} = \frac{1}{3N_q} \sum_{i=1}^{N_q} \left\| [u_i, v_i]^\top - [\bar{u}_i, \bar{v}_i]^\top \right\|_1, \end{aligned} \tag{2}$$

where 1/3 is a scaling factor to balance the effect of the projection loss. We consider a **separation loss** ($\mathcal{L}_{sep}$) that ensures that no more than one key-point can exist at the same 3D location. The loss penalizes two predicted key-points that are closer than hyperparameter δ^2 (0.05). It is calculated as:

$$\mathcal{L}_{sep} = \frac{1}{{N_q}^2} \sum_{i=1}^{N_q} \sum_{j \neq i}^{N_q} \max(0, \delta^2 - \left\| p_i - p_j \right\|_2^2), \tag{3}$$

where, p_i and p_j represent the i^{th} and j^{th} predicted key-point, respectively.

The point clouds based approaches predict key-points from the input point clouds and hence they do not consider object's surrounding. In contrast, an image based approach can predict 3D key-points in object's surroundings. To prevent this issue, a **shape consistency loss** ($\mathcal{L}_{shape}$) is included that forces the network to predict key-points closer to the surface of the object. It can be described as:

$$\begin{aligned} d_i = \left\| p_i - kNN(p_i, \mathcal{PC}) \right\|_2, \\ \mathcal{L}_{shape} = \frac{min(1, M)}{max(1, M)} \sum_{i=1}^{N_q} d_i, \quad \text{if} \;\; d_i > \gamma, \end{aligned} \tag{4}$$

where, $kNN()$ is a function that finds the nearest neighbor of a valid predicted key-point p_i from a point cloud $\mathcal{PC}$ of an object which is available during the

training time. The d_i is the distance of a key-point p_i from its nearest neighbor point while M is a count of considered distances that are greater than γ (0.05). $\mathcal{L}_{shape}$ is an average of the considered distances.

During training, we use ground truth information for the separation of valid key-points (from the predicted 21 key-points) for calculating the losses. However, during inference, to identify these valid key-points without ground truth information, a confidence score is required for every predicted key-point. For that, we compute a **confidence score loss** ($\mathcal{L}_{conf}$) by comparing predicted scores ($\mathcal{C}_{\mathcal{P}} = \{cp_i \mid i = 1, ..., N\}$) with ground truth scores ($\mathcal{C}_{\mathcal{Q}} = \{cq_i \mid i = 1, ..., N\}$) as;

$$\mathcal{L}_{conf} = \frac{1}{N} \sum_{i=1}^{N} \left\| cp_i - cq_i \right\|, \tag{5}$$

where cp_i could range from 0 to 1, whereas, cq_i could be either 1 or 0. The 1 and 0 represent if the key-point exists (is valid) for the object or not, respectively. N is the total number of predicted key-points, which is 21. Moreover, we pad zeros at the end of the ground truth score vector if it contains less than N scores. Where the zeros represent invalid key-points. The overall loss can be defined as:

$$\mathcal{L}_{overall} = \mathcal{L}_{pos} + \mathcal{L}_{proj} + \mathcal{L}_{sep} + \mathcal{L}_{shape} + \mathcal{L}_{conf}. \tag{6}$$

3.2 Inference

During inference, the network predicts 21 3D key-points along with their confidence scores from a single image. All the key-points having a confidence score greater than 0.5 are selected as valid key-points. The rest of the key-points are discarded. For better visualization, the predicted valid key-points are illustrated on the original point cloud of the object (i.e. Fig. 3).

3.3 Implementation details

The feature extractor module is based on ResNet-18 that is pre-trained on ImageNet dataset [36]. We discard its last two layers to extract features of dimensions 512x5x5. The network is implemented in PyTorch and trained with Adam optimizer. The learning rate is 10^{-3}, and the batch size is 512.

4 Experiments

The section presents an arrangement of the dataset, explains metrics selected for performance evaluation and compares the results with SOTA approaches.

4.1 Dataset

As extensively evaluated in previous approaches, we use KeypointNet dataset [3] to analyse the performance of our approach. It contains 8329 3D models, corresponding point clouds, and 83231 key-points of 16 object categories. However, it

does not contain images along with camera parameters that are required in our experiments. We render (RGB/RGBA) images in 24 different views by placing the object's 3D model at the origin of the reference frame and the virtual cameras at different locations, pointed towards the origin. During training, original point clouds along with the ground truth key-points are transformed w.r.t. the objects view in the input images. It allows the proposed approach to estimate key-points in different poses. We use the data split provided by KeypointNet.

4.2 Performance measurement

We compare our results with those of KP-Net [1]. Unlike the existing point cloud based methods [3, 22–25], their approach (in inference) uses single image and estimates 3D key-points (pixel $[u, v]$ and depth $[d]$). It estimates 3D key-points for two views of an object. The key-points are then used for finding a pose (rotation matrix) between the object views. The estimated pose is compared with the ground truth pose by computing an angular distance error.

We follow the same procedure and estimate key-points for two views (A and B) of an object using our approach. However, for evaluation, we use these key-points in two different methods. The first method is exactly the same as KP-Net, where we compute relative rotation matrix ($\bar{R}$) between object views using Procrustes analysis and then calculate the angular distance error (E_{rot_mat}) between computed and ground truth relative rotation matrix (R) as:

$$E_{rot\_mat} = 2\ arcsin\left(\frac{1}{2\sqrt{2}}\ ||\bar{R} - R||_F\right). \tag{7}$$

As a second evaluation, we transform the estimated key-points of view A ($\mathcal{A} = \{a_i | i = 1, ..., N\}$) using the predicted ($\bar{R}$) and the ground truth ($R$) rotation matrix and call them $\mathcal{A}_p = \{ap_i | i = 1, ..., N\}$ and $\mathcal{A}_q = \{aq_i | i = 1, ..., N\}$, respectively. Generally, both the key-points $\mathcal{A}_p$ and $\mathcal{A}_q$ should lie on the same positions as the key-points of view B (see Fig. 3). Every key-point ap_i/aq_i of $\mathcal{A}_p/\mathcal{A}_q$ is considered as a vector from the origin $(\mathbf{ap}_i, \mathbf{aq}_i)$. An angular distance error (E_{3D_pos}) between $\mathcal{A}_p$ and $\mathcal{A}_q$ is computed using vector dot product as:

$$E_{3D\_pos} = \frac{1}{N}\sum_{i=1}^{N} arccos\left(\frac{\mathbf{ap}_i \cdot \mathbf{aq}_i}{|\mathbf{ap}_i|\ |\mathbf{aq}_i|}\right), \tag{8}$$

where N is total number of estimated valid key-points. For a fair comparison with the KP-Net, we consider the first evaluation. Nevertheless, for validation on other categories, results from both evaluations are presented.

4.3 Results and Analysis

The proposed approach is evaluated using white background images of 13 different categories – 10 more than the KP-Net [1]. Two views of an object are passed to the network for estimating 3D key-points for every view. The Procrustes analysis

Table 1: Error in pose estimation between two views of an object. Angular distance error is computed in degrees between; 1) estimated and ground truth rotation matrices (Eq. 7) and 2) 3D positions (Eq. 8) of the predicted key-points in two views. MSE is computed between predicted and ground truth key-points.

Category	Error b/w Rot. matrices		Error b/w key-points 3D positions		MSE b/w predicted & ground truth key-points	
	Mean	Median	Mean	Median	Mean	STD
Airplane	6.581	3.145	5.963	2.565	0.006	0.017
Car	6.761	2.980	5.316	2.456	0.008	0.040
Chair	13.562	5.017	11.247	4.566	0.015	0.049
Table	23.919	3.635	18.079	2.975	0.053	0.159
Vessel	14.652	4.392	11.655	3.478	0.026	0.075
Bed	28.598	12.422	25.332	9.049	0.094	0.163
Cap	16.904	8.193	13.634	6.261	0.031	0.063
Helmet	26.947	16.058	23.504	15.243	0.062	0.076
Knife	25.330	13.006	20.599	12.490	0.008	0.006
Motorcycle	9.467	3.226	6.490	2.507	0.011	0.045
Guitar	19.559	5.289	7.247	2.926	0.003	0.006
Mug	18.470	9.135	10.320	5.942	0.026	0.056
Bottle	17.118	14.854	14.674	12.013	0.023	0.027
Average	16.962	7.690	12.822	6.190	0.028	0.060

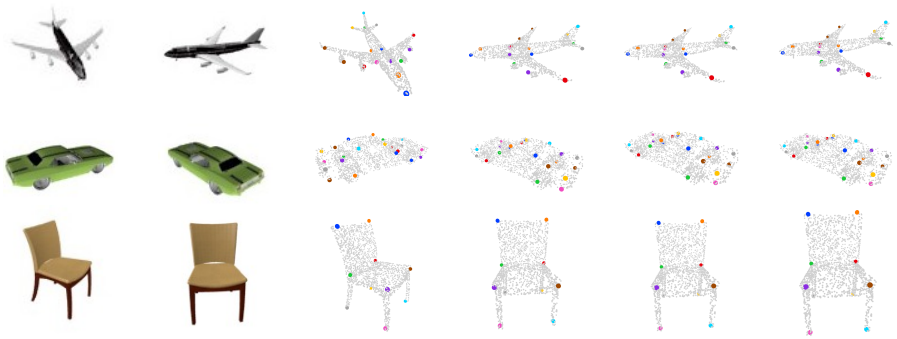
is used that utilizes the estimated key-points to compute a pose (rotation matrix) between the views. The error in the estimated pose is computed using both the evaluations (Eq. 7 and Eq. 8). The results are depicted in Tab. 1. The last two columns present the mean and Standard Deviation (STD) of the MSE between predicted and ground truth 3D key-points. The error is comparatively high for some categories. It is due to the structural variation (single/bunk beds, tables), different key-points for similar object shapes (helmet, knife, etc.), and differences in center of rotation and the center of mass of the object (i.e., mug). Qualitative results are given in supplementary material (**Sup. Fig. 1**).

To compare our results with the KP-Net, we consider the same three categories (cars, airplanes, and chairs) as reported in [1]. For the evaluation, we compute the angular distance error between the estimated and ground truth pose of two views of an object (see Eq. 7). The error is depicted in Tab. 2. In [1], the results are presented for four different versions; 1) supervised KP-Net that learns from ground truth 2D pixels and corresponding depths, 2) supervised KP-Net with a pretrained Orientation Network (O-Net) that provides an object's orientation information, 3) KP-Net (unsupervised) with O-Net, and 4) KP-Net without O-Net. It is reported in [1] that the KP-Net without O-Net performs overall well. The first four rows of the Tab. 2 present results of the KP-Net versions. In comparison to those, our results are more accurate.

Qualitative results are illustrated in Fig. 3. Columns (a) and (b) show two

Table 2: Error in pose estimation between two views of the same object. Mean and median angular distance errors are calculated (in degrees) between ground truth rotation and the rotation computed by Procrustes estimates between predicted key-points of the two views. Results of the baselines (first four rows) are the same as reported in [1]. All the results are produced for transparent images.

Method	Car		Airplane		Chair	
	Mean	Median	Mean	Median	Mean	Median
Supervised KP-Net	16.268	5.583	18.350	7.168	21.882	8.771
Supervised KP-Net with O-Net	13.961	4.475	17.800	6.802	20.502	8.261
KP-Net with O-Net	13.500	4.418	18.561	6.407	14.238	5.607
KP-Net	11.310	3.372	17.330	5.721	14.572	5.420
Ours	**5.190**	**2.073**	**3.257**	**2.053**	**10.732**	**4.096**



(a) View A (b) View B (c) KP A (d) KP B (e) KP A_{est} (f) KP A_{gt}

Fig. 3: Computing pose between two views (a) and (b) of an object. The corresponding estimated key-points are shown on the original point clouds in (c) and (d). The key-points of view A (c) are transformed to view B using estimated and ground truth rotation matrix as illustrated in (e) and (f), respectively.

views of the same object. The corresponding estimated key-points are presented in columns (c) and (d), respectively. Finally, the key-points (and point clouds) of view A after transformation using estimated (A_{est}) and the ground truth (A_{gt}) rotation are illustrated in (e) and (f). It can be visualized that the pose of the transformed key-points (e and f) is the same as the pose of key-points of view B (d). The experiment highlights that; 1) the estimated key-points can be used for computing a pose between two views, 2) the key-points are in semantical order, which is independent of the object view, and 3) the network can predict key-point of the occluded part of the object (i.e., back legs of the chair). Furthermore, we present another experiment that highlights the significance of the confidence score. We compare the predicted valid key-points (to whom the network assigns

Table 3: Comparison of the key-points predicted as valid by our network based on confidence scores (Pred.) with the key-points selected using ground truths (GT). The pose estimation error in two views of an object is approx. the same in both the cases; either the Pred. or GT key-points are used. Mean and SE of the pose error (calculated in both the methods using (a) rotation matrices (Eq. 7) and (b) key-points 3D positions (Eq. 8)) is given for RGB and RGBA images.

Category	Metric	Error b/w rotation matrices				Error b/w 3D positions			
		RGB		RGBA		RGB		RGBA	
		Pred.	GT	Pred.	GT	Pred.	GT	Pred.	GT
Airplane	Mean	6.581	6.552	3.257	3.267	5.963	5.974	2.805	2.797
	SE	0.195	0.194	0.075	0.076	0.003	0.003	0.001	0.001
Car	Mean	6.761	6.764	5.190	5.187	5.316	5.334	4.040	4.057
	SE	0.318	0.318	0.277	0.280	0.004	0.004	0.004	0.004
Chair	Mean	13.56	13.56	10.73	10.71	11.25	11.25	7.53	7.54
	SE	0.340	0.340	0.330	0.327	0.004	0.004	0.006	0.004

confidence greater than 0.5) with the key-points known to be present because of ground truths. The results are approx. the same in both cases, which validates that the confidence score helps the network in classifying the valid key-points for every object. The results are given in Tab. 3 that show mean angular distance error and the Standard Error (SE) which is calculated as $\sigma/\sqrt{n}$, where σ is the standard deviation of n angular distance errors. Additional results, including evaluation for realistic images, can be found in the supplementary material.

In a nutshell, it can be inferred that if a network could not estimate the confidence scores, it should predict fixed numbers of key-points as followed by the existing approaches. Otherwise, It may not be possible for the network to separate valid key-points from the total predicted N (21) key-points. Moreover, the confidence score allows jointly training a network for several categories with a different number of key-points. Otherwise, either the network can be trained for a single category, or the total key-points should be fixed for all the categories.

5 Conclusions

The paper presents an end-to-end solution for 3D key-points estimation from a single-view RGB image. The proposed approach extracts object features from an image, computes pixel-wise features by upsampling, and uses them for estimating 3D key-points along with confidence scores that reflect validity of the key-points. It enables the network to predict a different number of key-points based on the object shape. The key-points are estimated in an ordered semantic list, which increases its significance. Moreover, the network can be trained together for all the classes. The approach is evaluated by computing the pose between two views of an object. Our results are more accurate than those reported by KP-Net [1].

References

1. Suwajanakorn, S., Snavely, N., Tompson, J., Norouzi, M.: Discovery of latent 3d keypoints via end-to-end geometric reasoning. NeurIPS. (2018)
2. Spezialetti, R., Salti, S. and Di Stefano, L.: Performance Evaluation of 3D Descriptors Paired with Learned Keypoint Detectors. AI, 2(2), pp.229-243 (2021)
3. You, Y., Lou, Y., Li, C., Cheng, Z., Li, L., Ma, L., Lu, C., Wang, W.: Keypointnet: A large-scale 3d keypoint dataset aggregated from numerous human annotations. CVPR. pp. 13647-13656 (2020)
4. Bisio, I., Haleem, H., Garibotto, C., Lavagetto, F. and Sciarrone, A.: Performance Evaluation and Analysis of Drone-based Vehicle Detection Techniques From Deep Learning Perspective. IEEE Internet of Things Journal. 14(8), (2021)
5. Shu, Z., Yang, S., Xin, S., Pang, C., Jin, X., Kavan, L. and Liu, L.: Detecting 3D Points of Interest using Projective Neural Networks. IEEE Transactions on Multimedia. (2021)
6. Lin, Y., Chen, L., Huang, H., Ma, C., Han, X. and Cui, S.: Beyond Farthest Point Sampling in Point-Wise Analysis. arXiv preprint arXiv:2107.04291 (2021)
7. Zheng, Z., Yu, T., Dai, Q., Liu, Y.: Deep implicit templates for 3D shape representation. CVPR. pp. 1429-1439 (2021)
8. Zhao, W., Zhang, S., Guan, Z., Zhao, W., Peng, J., Fan, J.: Learning deep network for detecting 3d object keypoints and 6d poses. CVPR. pp. 14134-14142 (2020)
9. Liu, L., Yang, L., Chen, W., Gao, X.: Dual-view 3D human pose estimation without camera parameters for action recognition. IET Image Processing. (2021)
10. Tang, R., Wang, L., Guo, Z.: A Multi-Task Neural Network for Action Recognition with 3D Key-Points. ICPR. pp. 3899-3906. (2021)
11. Paoletti, G., Cavazza, J., Beyan, C. and Del Bue, A.: Unsupervised Human Action Recognition with Skeletal Graph Laplacian and Self-Supervised Viewpoints Invariance. BMVC, (2021)
12. Yuan, Y., Wei, S.E., Simon, T., Kitani, K., Saragih, J.: SimPoE: Simulated Character Control for 3D Human Pose Estimation. CVPR. pp. 7159-7169 (2021)
13. Wandt, B., Rudolph, M., Zell, P., Rhodin, H., Rosenhahn, B.: CanonPose: Self-supervised monocular 3D human pose estimation in the wild. CVPR. pp. 13294-13304 (2021)
14. Zhang, C., Zhan, F., Chang, Y.: Deep Monocular 3D Human Pose Estimation via Cascaded Dimension-Lifting. arXiv preprint arXiv:2104.03520. (2021)
15. Wan, C., Probst, T., Gool, L.V., Yao, A.: Self-supervised 3d hand pose estimation through training by fitting. CVPR. pp. 10853-10862 (2019)
16. Li, Y., Torralba, A., Anandkumar, A., Fox, D., Garg, A.: Causal discovery in physical systems from videos. arXiv preprint arXiv:2007.00631 (2020)
17. Paoletti, G., Cavazza, J., Beyan, C. and Del Bue, A.: January. Subspace Clustering for Action Recognition with Covariance Representations and Temporal Pruning. ICPR, pp. 6035-6042 (2021)
18. Shi, R., Xue, Z., You, Y., Lu, C.: Skeleton Merger: an Unsupervised Aligned Keypoint Detector. CVPR. pp. 43-52 (2021)
19. You, Y., Liu, W., Li, Y.L., Wang, W., Lu, C.: UKPGAN: Unsupervised KeyPoint GANeration. arXiv preprint arXiv:2011.11974 (2020)
20. BOJANIĆ, D., BARTOL, K., PETKOVIĆ, T., PRIBANIĆ, T.: A Review of rigid 3D registration methods. 13th International Scientific-Professional Symposium Textile Science and Economy. pp. 286–296 (2020)

21. Adamczyk, D., Hula, J.: Keypoints Selection Using Evolutionary Algorithms. In ITAT. pp. 186-191 (2020)
22. Fernandez-Labrador, C., Chhatkuli, A., Paudel, D.P., Guerrero, J.J., Demonceaux, C., Gool, L.V.: Unsupervised learning of category-specific symmetric 3D keypoints from point sets. ECCV. pp. 546-563 (2020)
23. Jakab, T., Tucker, R., Makadia, A., Wu, J., Snavely, N., Kanazawa, A.: KeypointDeformer: Unsupervised 3D Keypoint Discovery for Shape Control. CVPR. pp. 12783-12792 (2021)
24. Chen, N., Liu, L., Cui, Z., Chen, R., Ceylan, D., Tu, C., Wang, W.: Unsupervised learning of intrinsic structural representation points. CVPR. pp. 9121-9130 (2020)
25. Wu, W., Zhang, Y., Wang, D., Lei, Y.: SK-Net: Deep learning on point cloud via end-to-end discovery of spatial keypoints. AAAI. 34(04), pp. 6422-6429 (2020)
26. Georgakis, G., Karanam, S., Wu, Z., Kosecka, J.: Learning local rgb-to-cad correspondences for object pose estimation. ICCV. pp. 8967-8976 (2019)
27. You, Y., Li, C., Lou, Y., Cheng, Z., Li, L., Ma, L., Wang, W., Lu, C.: Fine-grained Object Semantic Understanding from Correspondences. arXiv preprint arXiv:1912.12577 (2019)
28. He, Y., Sun, W., Huang, H., Liu, J., Fan, H., Sun, J.: Pvn3d: A deep point-wise 3D keypoints voting network for 6dof pose estimation. CVPR. pp. 11632-11641 (2020)
29. Wang, C., Martín-Martín, R., Xu, D., Lv, J., Lu, C., Fei-Fei, L., Savarese, S., Zhu, Y.: 6-PACK: Category-level 6d pose tracker with anchor-based keypoints. ICRA. pp. 10059-10066 (2020)
30. Devgon, S., Ichnowski, J., Balakrishna, A., Zhang, H., Goldberg, K.: Orienting Novel 3D Objects Using Self-Supervised Learning of Rotation Transforms. IEEE 16th International Conference on Automation Science and Engineering (CASE). pp. 1453-1460 (2020)
31. Sun, W., Tagliasacchi, A., Deng, B., Sabour, S., Yazdani, S., Hinton, G., Yi, K.M.: Canonical capsules: Unsupervised capsules in canonical pose. arXiv preprint arXiv:2012.04718. (2020)
32. Barabanau, I., Artemov, A., Burnaev, E., Murashkin, V.: Monocular 3d object detection via geometric reasoning on keypoints. arXiv preprint arXiv:1905.05618 (2019)
33. Lu, J., Richter, F., Yip, M.: Robust keypoint detection and pose estimation of robot manipulators with self-occlusions via sim-to-real transfer. arXiv preprint arXiv:2010.08054 (2020)
34. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM. 24(6), pp.381-395 (1981)
35. Han, K., Rezende, R.S., Ham, B., Wong, K.Y.K., Cho, M., Schmid, C., Ponce, J.: Scnet: Learning semantic correspondence. ICCV. pp. 1831-1840 (2017)
36. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. CVPR. pp. 248-255 (2009)