Abstract—This paper introduces a novel 6D pose estimation algorithm – $\mathcal{W}$-PoseNet, which directly regresses object rotation and translation from a sparse set of pixel pair representations, via a low-rank bilinear pooling on dense features of input RGB-D images. Moreover, those pixel-wise deep features are regularized by explicitly learning a dense correspondence mapping onto their 3D coordinates in a canonical space as an auxiliary task, which can thus improve robustness against ambiguities caused by pose symmetries and inter-object occlusion. Experiment results on two popular benchmarks show that our $\mathcal{W}$-PoseNet consistently achieves state-of-the-art performance on 6D pose estimation.

I. INTRODUCTION

The problem of six degree-of-freedom pose (simply put, 6D pose) estimation aims to predicting a rotation together with a translation of an object instance in 3D space relative to a canonical CAD model, which plays a vital role in a number of applications such as augmented reality [1], [2], grasp and manipulation in robotics [3], [4], [5], and 3D semantic analysis [6], [7], [8], [9]. This paper concerns on the problem of estimating 6D pose of an object given a RGB-D image. Such a problem remains challenging in view of intrinsic inconsistent texture and shape of objects and inter-object occlusion in the cluttered scenes, in addition to extrinsic varying illumination and sensor noises. In light of this, encoding a discriminative and robust feature representation for each instance is essential for predicting its 6D pose.

Most of existing RGB-D methods [10], [11], [12] depend on iterative-closest point (ICP [13]) to refine pose predictions, which leads to less efficient inference compared to learning-based refinement (such as the refinement network adopted in [14] achieving hundreds of magnitude faster) and thus can be less favorable for real-time applications such as robotic grasp and planning. DenseFusion [14] has recently been developed to combine both textural and geometric features in a pixel-wise fusion manner, which can obtain accurate pose estimation in a real-time inference processing. Cheng et al. [15] further exploit both intra-modality and inter-modality correlation to learn more discriminative local feature based on DenseFusion [14]. However, these state-of-the-art dense pose regressors heavily depend on the quality and resolution of the acquired data and also suffer from pixel-wise feature ambiguities caused by pose symmetries.

We consider that extracting discriminative textural and geometric features in local regions and generating a robust global representation of each object instance are important to alleviate the aforementioned challenges in 6D pose estimation. To this end, different from pixel-wise regression methods such as DenseFusion [14] or Correlation Fusion [15], this paper is the first attempt to explore pixel-pair pose regression – $\mathcal{W}$-PoseNet by developing two novel components of in the proposed deep model: a joint loss function and a $\mathcal{W}$-shape pose regression module, which is shown in Fig. 1.

On one hand, beyond one loss term as other pixel-wise pose regressors [14], [15] on minimizing pose predictions and their corresponding ground truth, the proposed $\mathcal{W}$-PoseNet additionally encodes the textural and geometric information favoring for reconstructing pose-sensitive point sets, whose points’ 3D coordinates are generated by transforming the observed point cloud sampling from object models (e.g. CAD models) with the inverse of ground truth pose. In detail, our $\mathcal{W}$-PoseNet incorporates an auxiliary task of dense correspondence mapping from each pixel of input...
data to its corresponding coordinate in the canonical space, in order to regularize local feature learning for pose regression by extra pixel-wise supervision self-generated from object models and ground truth pose. Intuitively, compared to sparse semantic keypoints pre-defined in keypoint-based methods [16], [17], [18], [19], [20], dense correspondence mapping in our scheme treats each pixel as a keypoint to regress its corresponding 3D coordinate in object model space, which makes each pixel-wise feature more discriminative, and thus our pixel-wise feature encoding is more robust to occlusion.

On the other hand, inspired by the point-pair feature [21], [22] on point clouds, this paper proposes a novel pixel pair feature encoding layer based on the low-rank bilinear pooling [23], [24] to generate pixel-pair features from two sampled pixels’ vectors in feature map output of the encoder, which are then mapped onto 6-DoF pose. Similar to the concept in [21], [22], our pixel-pair features describe the relative geometric structure of two sampled 3D points on the object models, but also contain texture information of their 2D projection on input RGB images. Moreover, local features anchored on single pixels concern on extracting textural and geometric information in local regions, which can be less discriminative to some symmetric poses. Such an observation encourages our motivation to combine pixel-wise features into a set of pixel-pair features. A global representation for each instance, which combines a number of pixel-pairs’ features, thus generates a pose prediction.

The main contributions of this paper are three-fold.

• This paper is the first attempt to introduce the concept of a pair-wise combination of dense features to 6D pose estimation. To this end, this paper proposes a novel pixel-pair pose regression network – \(W\)-PoseNet, which aims to learn a global representation of each object instance consisting of sparse pixel-pair features (PiPF).

• This paper designs an auxiliary task of dense correspondence mapping from input data to 3D coordinates in object model space sensitive to 6D pose, which can regularize feature encoding in deep pose regression to improve pixel-wise local feature discrimination.

• Extensive experiments are conducted on the popular benchmarks, whose results can demonstrate that the proposed \(W\)-PoseNet consistently outperforms the state-of-the-art 6D pose estimation algorithms.

Source codes and pre-trained models will be released after acceptance at \(https://github.com/xzlsanut/W-PoseNet\).

II. RELATED WORKS

Keypoint-based 6D Pose Estimation – The algorithms belonging to keypoint-based pose estimation share a two-stage pipeline: first localizing 2D projection of predefined 3D keypoints and then generate pose predictions via 2D-to-3D keypoint correspondence mapping with a PnP [26]. Existing methods can be categorized into two groups: object detection based [19], [27] and dense heatmap based [17], [18]. The former focuses on solving the problem of sparse keypoint localization via object detection on the whole object to reduce negative effects of background [19], [27], but are sensitive to occlusion [17]. The latter group of methods [17], [18] pay more attention to discovering latent correlation across all the keypoints and thus are more robust to inter-object occlusion. Recently, PVNet [20] is proposed to detect keypoints via voting on pixel-wise predictions of the directional vector that points to keypoints and is robust to truncation and occlusion. In contrast, dense correspondence mapping in our method share similar concepts as keypoint-based methods but makes pixel-wise predictions on 3D keypoints directly instead of their projection in 2D images, which performs robustly to handle with occlusion.

Dense 6D Pose Estimation – An alternative group of algorithms is to produce dense pose predictions for each pixel or local patches with hand-crafted features [28], [29], CNN patch-based feature encoding [30], [8] and CNN pixel-based feature encoding [14], [15], whose final pose output is selected via a voting scheme. In [31], [32], random forests are adopted to regress 3D coordinates for each pixel and discover 2D-3D correspondence to produce pose predictions. The proposed \(W\)-PoseNet is designed to both predict poses from dense features and generate pose-sensitive coordinates in a unique framework. In general, the dense feature encoding module in our method follows the same pipeline as [14], [15], but the key difference lies in incorporating an extra branch to constrain feature encoding with dense correspondence mapping from input data to its corresponding 3D coordinates in a canonical space.

Bilinear Pooling in CNNs – The bilinear pooling [33] is an effective tool to calculate second-order statistics of local features in visual recognition but suffers from expensive computational cost due to its high-dimensionality. A number of its variants have been proposed to cope with the challenge via approximation with Random Maclaurin [34] or Tensor Sketch [35]. A low-rank bilinear model is proposed by Kong et al. [23] to avoid computing the bilinear feature matrix, which is approximated via two decomposed low-rank matrices. Similarly, [24] introduces Grassmann Pooling based on SVD decomposition to approximate feature maps. A quadratic transformation with the low-rank constraint is exploited on pairwise feature interaction either from spatial locations [36] or from feature maps across network layers [37]. Our pixel-pair pose regression shares a similar concept as bilinear models to construct pair-wise features, but a sparsely sampled set of pixel-pair features in \(W\)-PoseNet can achieve more robust estimation, owing to the pair-wise combination of spatially localized features.

III. METHODOLOGY

The problem of 6D pose estimation given RGB-D images is to detect object instances in the scenes and estimate their rotation \(R \in SO(3)\) and translation \(t \in \mathbb{R}^3\). In detail, a 6D pose can be defined as a rigid transformation \(p = [R|t]\) from the object coordinate system with respect to
the camera coordinate system. This paper aims to improve the discrimination and robustness of a global representation of each object instance having large variations of texture and shape, from the perspective of pairwise feature interaction on dense local features.

Fig. 2 illustrates the whole pipeline of our $W$-PoseNet, which consists of several main stages in addition to iterative pose refinement as DenseFusion [14] (see Sec. III-D). Encouraged by the recent success of dense pose regression on RGB-D data, we employ the same feature encoding and fusion part of DenseFusion [14] to extract and fuse pixel-wise features from heterogeneous data (see Sec. III-A). Pixel-wise features are sampled to generate a sparse set of pixel-pair features via a low-rank bilinear pooling on local features, which produces a global feature via a max pooling and its pose prediction (see Sec. III-B). For robust pixel-wise features, an auxiliary task of learning a dense correspondence mapping from each pixel to its 3D coordinate together with features here is to get rid of the global information about neighboring features. Our motivation for not using global features here is to get rid of the global information about object instances to verify the effectiveness of a combination of local features as a global description.

A. Semantic Segmentation and Pixel-Wise Feature Encoding

We adopt the identical modules of the DenseFusion [14] and briefly introduce the following steps: 1) semantic segmentation on RGB images and point clouds converted by depth images; 2) dense feature extraction and 3) pixel-wise feature fusion, which is shown in the top row of Fig. 2.

Following the segmentation algorithm adopted in [14], [10], an RGB image is first fed into the autoencoder-based segmentation network to produce $N + 1$ binary masks belonging to $N$ object classes and the background class respectively. The bounding box of each mask is used to crop the corresponding depth image, and the depth intensities within the object mask are converted to a point cloud.

Cropped image patches and point clouds are fed into 2D CNN based and PointNet [38] based feature encoders to respectively extract texture and geometric features from heterogeneous data sources. Specifically, given $H \times W$ RGB-D images, the texture branch aims to mapping $H \times W \times 3$ images to $H \times W \times d_{\text{rgb}}$-dimensional feature maps, which correspond to a feature vector $f \in \mathbb{R}^{d_{\text{rgb}}}$ at each spatially localized pixel. The other branch extracts geometric features $g$ from orderless points via two shared-MLP and produces $P \times d_{\text{depth}}$-dimensional feature maps where $P$ denotes the size of sampled points in point cloud generated from depth images.

We combine textural features $f$ from those pixels in cropped RGB image patches corresponding to these $P$ points with $g$ in the manner of pixel-wise fusion. Specifically speaking, to capture correlation across RGB and depth modalities, geometric and appearance features belonging to the same pixels are first concatenated and then fed into a MLP-based feature fusion network (i.e. PointNet++ [25]) to generate $P$ pixel-wise fusion feature $x \in \mathbb{R}^{d_{\text{fusion}}}$. Instead of concatenating the fused feature with a global feature obtained by average pooling as in the DenseFusion [14], we adopted the hierarchical pooling of PointNet++ [25] to aggregate local neighboring features. Our motivation for not using global features here is to get rid of the global information about object instances to verify the effectiveness of a combination of local features as a global description.

B. Pixel Pair Pose Regression

This section presents the key components of our pixel-pair pose prediction: 1) pixel-pair feature generation; 2) pixel-pair feature encoding; and 3) pose regression, which are shown in the bottom row of Fig. 2.

The seminal work of point-pair feature (PPF) [21] constructs 4-dimensional features of a relative position in the manner of Euclidean distance and relative angles between normals and between normals and the determined line from a pair of oriented points sampled from an observed 3D scene. Motivated from its elegant design that can precisely determine a pose when the pair of points is sampled from the object of interest, we propose to pair-wisely aggregate the learned pixel-wise features for a pixel-pair feature encoding (PiPE) to alleviate the suffering of dense features from
In view of high-dimensionality matrices, the computational simplicity of direct concatenation, which reveals the 1-order statistical information of local features. An alternative to exploit 1-order statistical information is to generate an element-wise sum of two feature vectors. Encouraged by bilinear models in visual recognition [34], [35], [24], our pixel-pair feature encoding is designed based on a low-rank bilinear pooling [39]. Specifically, given two pixel-wise feature vectors (e.g. \( a \) and \( b \) from the set \( \{ x \}^P \)), the second-order pooling method [40] can be written as

\[
G_{\text{PIPE}}(a, b) = \text{vec}(ab^T) \in \mathbb{R}^{d_{\text{fusion}}^2},
\]

In view of high-dimensionality pooling of \( ab^T \), the low-rank bilinear pooling [39] can be adopted to obtain two low-rank matrices \( U \in \mathbb{R}^{d_{\text{fusion}} \times 1} \) and \( V \in \mathbb{R}^{d_{\text{fusion}} \times 1} \) to avoid computing \( ab^T \) as

\[
G_{\text{PIPE}}(a, b) = \hat{P}^T \sigma(U^Ta \circ V^Tb)
\]

where \( \circ \) denotes the Hadamard product, \( \hat{P} \) is the project matrix to determine the output dimension, and we follow [39] use the relu nonlinearity activation function \( \sigma(\cdot) \). For generating a \( d_{\text{fusion}} \)-dimensional pixel-pair feature (the setting in our experiments for a fair comparison), we compare computational complexities of direct concatenation and projection via a fully-connected layer (\( 2 \times d_{\text{fusion}} \), \( d_{\text{fusion}} \)) and a low-rank bilinear pooling. The former desires \( 2 \times d_{\text{fusion}} \) network parameters, while the latter needs \( 3 \times d_{\text{fusion}} \times l \) where \( l \) denoting the dimension of the compressed vector projected from \( x \) is usually much smaller than \( d_{\text{fusion}} \). As a result, our method adopts the low-rank bilinear pooling in view of the lighter computational cost and the more richer information than direct concatenation, which is verified in our experiments.

**Pixel-Pair Feature Encoding (PIPE)** – Given the output of pixel pairs, a naive approach to combine local features (i.e., \( x_k \) where \( k = 1, 2, \ldots, P \)) into a compact one is direct concatenation, which reveals the 1-order statistical information of local features. In view of this, multiple point-pair features [21] consisting of relative position and orientation of two oriented points, are collected as a global representation of an object model. Such a feature representation can still perform robustly to occlusion and sparse point clouds, which is widely adopted for practical applications. Inspired by the robustness and low computational cost of point-pair features [21], we introduce a novel feature encoding layer inserted between the pose regression module and feature fusion module investigated in the previous section. Specifically, given the set \( \{ x \}^P \) of \( P \) pixel-wise features, we replicate the features into another set. Each feature vector in one set is combined with one randomly selected from the other set, then the \( P \) pixels will eventually generate \( P \) pixel-pairs. Other combination strategies such as dense pair generation by combining any two pixels can also be employed, but we adopt such a sparse generation owing to its computational simplicity.

**Pixel-Pair Pose Regression** – All \( P \) pixel-pair features \( G_{\text{ppf}} \) are fed into a max pooling to generate a global representation for 6D pose prediction of each object instance. Specifically, the error of pose predictions of the global representation as follows [14]:

\[
L_{\text{PR}} = \frac{1}{M} \sum_j \| \mathbf{R}x_j + t - (\hat{\mathbf{R}}x_j + \hat{t}) \|, \quad (3)
\]

where \( x_j \) denotes the \( j \)-th point in a point set randomly sampled from the object model, \( M \) is the size of the point set, \( [\mathbf{R}|t] \) is the ground-truth pose and \( [\hat{\mathbf{R}}|\hat{t}] \) is the predicted pose output by the pixel-pair feature.

**C. Auxiliary Dense Correspondence Mapping**

For obtaining discriminative and robust features per pixels, our method designs an auxiliary dense correspondence mapping task to regularize feature learning. As shown in Fig. 3, dense correspondence mapping is achieved via regressing per-pixel features \( x \) to its corresponding 3D coordinate in the canonical coordinate system. To this goal, the object point cloud obtained by a depth input image is firstly transformed with the inverse of ground truth pose \( [\mathbf{R}|t] \) to obtain its corresponding point cloud in the canonical space. Each 3D coordinate in the transformed point cloud is used as a pixel-wise supervision of its corresponding point in the dense correspondence mapping branch. The loss term on this auxiliary branch, termed as a Dense Correspondence Mapping (DCM) Loss, can be thus written as

\[
L_{\text{DCM}} = \frac{1}{M} \sum_i ||p_i - \hat{p}_i|| \quad (4)
\]

where \( M \) represents the points sampled from the observed point cloud, \( \hat{p} \) is the prediction of the DCM branch of our network, and \( p \) denotes the ground truth coordinate corresponding to \( \hat{p} \). A joint loss \( L_{\text{joint}} \) to combine both terms for pose regression and dense correspondence mapping can thus be written as:

\[
L_{\text{joint}} = L_{\text{PR}} + \lambda L_{\text{DCM}} \quad (5)
\]

where \( \lambda \) is the trade-off parameter between two terms.
D. Iterative Pose Refinement

Iterative refinement in the DenseFusion [14] has gained significant performance improvement for 6D pose estimation. Similarly, we adopt its identical refinement post-processing to improve final pose predictions, which encodes the resulting point clouds after transformation with previous pose predictions to gradually refining pose in a residual learning manner. Experimental results can demonstrate the boosted performance of the proposed \( \mathcal{W} \)-PoseNet together with the refinement network.

IV. EXPERIMENTS

Datasets – To evaluate \( \mathcal{W} \)-PoseNet comprehensively, we conduct experiments on two popular benchmarks – the YCB-Video dataset [10] and the LineMOD dataset [41]. The YCB-Video dataset consists of 21 objects with different textures and sizes and has 92 videos in total. Following recent work [10], [14], [15], we adopt 80 videos for training, 2949 keyframes from the other 12 videos for testing, and also used an additional 80,000 synthetic images provided by [10]. For a fair comparison, we used the semantic segmentation mask provided by the PoseCNN in evaluation by following [10], [14]. The LineMOD dataset contains 15,783 images belonging to 13 low-textural objects placed under different cluttered environments suffering from the challenges of occlusion and illumination changes. We follow the prior works [14], [15] use 15% of images for training and use the other 85% of images for testing without using additional synthetic data.

Performance Metrics – We adopt the ADD(S) and ADD(D) metrics in the YCB-Video dataset following recent work [10], [14] for comparative evaluation. For asymmetric objects, we use the average distance (ADD) between 3D model points transformed by ground-truth poses and predicted poses to measure the pose estimation error. For symmetric objects, the average closest point distance (ADDS) is employed to measure the mean error. 6d pose predictions are considered to be correct if the error is smaller than a predefined threshold, which varies from 0 to 10cm to plot an accuracy-threshold curve to compute the area under the curve (AUC). Similarly, for the LineMOD dataset, we use the ADDS distance for symmetric objects (i.e. eggbox and glue) and the ADD distance for the remaining objects having an asymmetric geometry while taking 10% of the diameter as threshold following [14], [19].

Implementation Details – As the DenseFusion [14], in our experiments, the CNN used for texture feature encoding is composed of Resnet-18 [42] followed by 4 blocks of one upsampling and one convolution layer as the decoder. RGB and depth images are respectively encoded into 128-dimensional vectors, which are fused into a pixel-wise dense feature at fusion. The PointNet++ [25] aggregating pixel-wise features consists of two set abstraction and two feature propagation layers. For training our model, learning rate is set \( 10^{-3} \) with the Adam optimizer.

A. Comparison with State-of-the-art Methods

We compare the proposed \( \mathcal{W} \)-PoseNet and the state-of-the-art methods on the YCB-Video and the LineMOD datasets, and the results are visualized in Tables I and II. In general, our method can consistently achieve state-of-the-art performance with and without iterative pose refinement on the YCB-Video and LineMOD. Specifically, on the YCB-Video, our network consistently performs better than its direct competitor DenseFusion especially for asymmetric objects with and without post-processing refinement, while similar results are observed on the LineMOD dataset. As the dense feature extraction and fusion part of our \( \mathcal{W} \)-PoseNet and DenseFusion are identical, the performance gain can only be explained by our design on a pixel-pair combination of local features and auxiliary dense correspondence mapping.

To verify the robustness of \( \mathcal{W} \)-PoseNet against inter-object occlusion, we first calculate the invisible surface percentage for each object instance on the YCB-Video dataset as the DenseFusion [14] did, then evaluate the proposed \( \mathcal{W} \)-PoseNet with different degrees of occlusion. Specifically, we sampled a certain number of points on object models, projected these points onto their image plane by using the ground-truth poses and the camera intrinsic parameters. Given the observation (i.e. depth \( d(p) \) on each pixel) in depth images and 2D projection \( \hat{d}(\tilde{p}) \) of 3D coordinates, if any pixel satisfying \( |d(p) - \hat{d}(\tilde{p})| > h \) (\( h \) is set to 2cm in our experiment), the pixel is considered under occlusion and can be marked as an invisible pixel. The invisible surface percentage for each instance can be generated via the ratio between the size of invisible points and the total number of sampled points. Fig. 4 shows comparative evaluation of several methods in terms of the ADD(S) metric. Our \( \mathcal{W} \)-PoseNet can consistently outperform the PoseCNN+ICP [10] and the DenseFusion [14] with increasing the invisible surface percentage. On one hand, both \( \mathcal{W} \)-PoseNet and DenseFusion are more robust than the PoseCNN+ICP against inter-object occlusion in view of dense feature encoding. On the other hand, the performance gain of our \( \mathcal{W} \)-PoseNet sharing the identical feature encoding module of the DenseFusion can demonstrate its superior robustness of our dense correspondence regularized pixel pair pose regression to pixel-wise regression in the DenseFusion.
B. Ablation Studies

In order to verify the effectiveness of each module of \( W \)-PoseNet, we conducted a number of ablation experiments on the LineMOD dataset without iterative pose refinement.

Effects of Pixel-Pair Pose Prediction – Compared to pixel-wise pose estimation in the DenseFusion [14], our \( W \)-PoseNet without the DCM Loss and post-refining gain 88.2% on the mean ADD(S), while the DenseFusion without refinement can only achieve 86.2%. The only difference between \( W \)-PoseNet without the DCM Loss and the DenseFusion lies in the usage of pixel-pair pose regression, and thus 2% improvement gain can be credited to the effects of Pixel-Pair Pose Prediction.

Evaluation on the DCM Loss – We compare the DenseFusion and our \( W \)-PoseNet without pixel-pair regression, obtaining 86.2% and 93.2% respectively, which only differ on the usage of additional dense correspondence mapping branch. It is evident that, owing to introducing the auxiliary DCM branch, the \( W \)-PoseNet can significantly outperform the state-of-the-art DenseFusion by 7%, which indicates that the DCM branch improves the quality of per-pixel features.

Effects of the Feature Fusion Module – We explore a variety of feature fusion methods, including the element-wise sum (ELS), direct concatenation (CON), and low-rank bilinear pooling (LRBP) under two settings, i.e. with and without the auxiliary correspondence mapping (DCM). Among these methods, the LRBP consistently achieves the best performance in both settings. Such a study verifies the efficacy of a bilinear pooling to exploit the richer 2-order statistics of local features.

Effects of Iterative Pose Refinement – We employ the identical refinement network with the same settings as the DenseFusion. From Tables [I] and [II], iterative pose refinement can further boost estimation performance of our \( W \)-PoseNet.

V. Conclusions

This paper introduces a novel 6D pose estimation network based on two key observations: extracting discriminative dense features in local regions and generating a robust global representation of each object instance. To this end, we introduce two novel modules into the state-of-the-art DenseFusion – pixel-pair pose prediction and dense correspondence mapping utilizing object models, which are verified their effectiveness in the experiments respectively.
