*electronics*

MDPI

*Article*

# FusionNet: An End-to-End Hybrid Model for 6D Object Pose Estimation

**Yuning Ye [1] and Hanhoon Park [1,2,*]**

[1] Department of Artificial Intelligence Convergence, Graduate School, Pukyong National University, 45 Yongso-ro, Nam-gu, Busan 48513, Republic of Korea; yeyuning12@gmail.com
[2] Division of Electronics and Communications Engineering, Pukyong National University, 45 Yongso-ro, Nam-gu, Busan 48513, Republic of Korea
* Correspondence: hanhoon.park@pknu.ac.kr; Tel.: +82-51-629-6225

**Abstract:** In this study, we propose a hybrid model for Perspective-n-Point (PnP)-based 6D object pose estimation called FusionNet that takes advantage of convolutional neural networks (CNN) and Transformers. CNN is an effective and potential tool for feature extraction, which is considered the most popular architecture. However, CNN has difficulty in capturing long-range dependencies between features, and most CNN-based models for 6D object pose estimation are bulky and heavy. To address these problems, we propose a lighter-weight CNN building block with attention, design a Transformer-based global dependency encoder, and integrate them into a single model. Our model is able to extract dense 2D–3D point correspondences more accurately while significantly reducing the number of model parameters. Followed with a PnP header that replaces the PnP algorithm for general end-to-end pose estimation, our model showed better or highly competitive performance in pose estimation compared with other state-of-the-art models in experiments on the LINEMOD dataset.

**Keywords:** object pose estimation; convolutional neural network; transformer; hybrid model; deep learning



**Citation:** Ye, Y.; Park, H. FusionNet: An End-to-End Hybrid Model for 6D Object Pose Estimation. *Electronics* **2023**, *12*, 4162. https://doi.org/10.3390/electronics12194162

Academic Editor: Zhenhua Guo

Received: 23 August 2023
Revised: 5 October 2023
Accepted: 6 October 2023
Published: 7 October 2023



## 1. Introduction

In augmented reality applications, estimating the 6D pose (3D rotation and translation) of objects with respect to the camera is a fundamental task. This task, such as many other vision tasks, witnessed a complete renaissance with the advent of deep learning. The classical approach for reliable 6D pose estimation is based on finding 2D–3D feature point pairs from input images, followed by the Perspective-n-Point (PnP) algorithm [1] to predict the object pose. The key to this approach is to extract informative features that facilitate pose estimation using well-designed convolutional neural networks (CNN) [2–4]. The accuracy of pose estimation depends heavily on the performance of CNNs. CNNs are the basic building blocks of deep learning models for vision tasks. The strength of CNNs lies in their ability to learn local spatial features. Building deep learning models using CNNs has become a common and dominant method for all kinds of vision tasks [5,6]. However, since convolution is an operation that deals with one local neighborhood at a time, CNN-based models usually suffer from capturing long-range dependencies. Capturing long-range dependencies is of central importance in deep neural networks, which helps global understanding of visual scenes [7,8]. There is no doubt that long-range dependencies also play an important role in estimating object poses. Commonly, long-range dependencies are modeled by the large receptive fields formed by deep stacks of convolutional operations. However, this approach still provides a weak and limited understanding of global features, rather making networks cumbersome, difficult to run on resource-constrained environments such as mobile or embedded systems. Although previous studies have attempted to overcome this problem [9,10], the performance has been validated for only a few limited vision tasks.

Transformer is an architecture based on a self-attention mechanism that has achieved state-of-the-art results in many natural language processing (NLP) tasks [11,12]. Recently, Transformer has also been extremely active in the field of computer vision. The Vision Transformer (ViT) is the first Transformer-based model for vision tasks that relies exclusively on the Transformer architecture, which aims to adapt the Transformer architecture with minimal modifications [13]. The input images are split into discrete non-overlapping patches, the patches are treated as markers (similar to tokens in NLP), summed up with positional encoding vectors to incorporate spatial information, and input into repeated Transformer layers to model global relationships for classification. Due to its excellent ability to model long-range dependencies, ViT has obtained competitive performance in large-scale image classification compared to CNN-based models, which has recently been extended and utilized in various vision tasks [14,15].

Despite the success of ViT and its variants, the performance is still lower than that of CNNs of similar size when trained on small amounts of data. One possible reason is that ViT lacks certain properties (usually known as inductive bias) that are inherent in CNNs and make CNNs well-suited for solving vision tasks; Images have a strong two-dimensional local structure and spatially adjacent pixels are usually highly correlated. CNN architectures force this local structure to be captured by using local receptive fields, shared weights, and spatial subsampling, and thus also achieve some degree of shift, scale, and distortion invariance [16,17].

To address the above limitations, inspired by previous studies [9,10,18,19], we hypothesize that Transformer architecture can be strategically introduced to the convolutional structure to improve performance and robustness, while concurrently maintaining a high degree of computational and memory efficiency. To verify the hypothesis, we present FusionNet, a novel architecture for PnP-based 6D object pose estimation. Similar to the state-of-the-art model [20], it takes RGB images as the input and estimates object poses in an end-to-end manner. However, FusionNet uses modified CNN blocks to extract informative features efficiently and incorporates Transformer blocks to capture long-range dependencies between features. Specifically, FusionNet has the following structural properties: First, the convolutional operations are partitioned into multiple stages that form a hierarchical structure. The hierarchical design helps to extract multi-scale features and reduces the computational burden associated with high resolution. Unlike ResNet blocks [21,22] commonly used in previous 6D object pose estimation studies, FusionNet has a newly designed CNN building block consisting of a convolutional block and an attention block. The convolutional block consists of $3 \times 3$ and $1 \times 1$ convolutional layers to keep the model lightweight while enhancing the ability to handle nonlinear features. The attention block helps to learn global context within intermediate features, reducing unnecessary computation and improving the representation ability of the network. Second, global dependency encoder (GDE), a Transformer, is introduced, and it receives features extracted from CNN blocks and encodes long-range dependencies. The encoder helps FusionNet to learn global features without making the CNN blocks deeper and wider, which allows for the light weight of FusionNet despite the fusion of two kinds of architectures (see Figure 1).

The primary contributions of this study focusing on developing an end-to-end hybrid model for 6D object pose estimation are as follows:

- We propose FusionNet, the first hybrid model designed specifically for 6D object pose estimation with the introduction of a CNN-based transformer. FusionNet takes advantages of both architectures: CNN and Transformer.
- We design an efficient CNN building block with self-attention that is lightweight but has excellent performance in extracting informative features for pose estimation.
- We also design a Transformer called GDE to encode long-range dependencies between local features.
- FusionNet is a lightweight model that is more suitable for resource-constrained devices common in real-life conditions than other 6D object pose estimation models. It can be easily implanted into mobile or embedded devices.

- FusionNet is flexible enough to be applied to other vision tasks requiring the ability of extracting informative features and capturing their long-range dependencies with minor modifications.
- The performance of FusionNet is validated on a benchmark dataset in various aspects. The experiments show that FusionNet outperforms other 6D object pose estimation models.

We note that research has been conducted such as CvT [17] and CMT [23] merging CNN and Transformer to leverage the advantages of both architectures. Recently, attempts of merging CNN and Transformer to solve low-level and high-level vision problems have also been reported [24–27]. However, to the best of our knowledge, there have been no attempts to merge both architectures in 6D object pose estimation, except by simply using CNNs at a pre-processing step for extracting features used as the input in Transformer-based models [28–31].

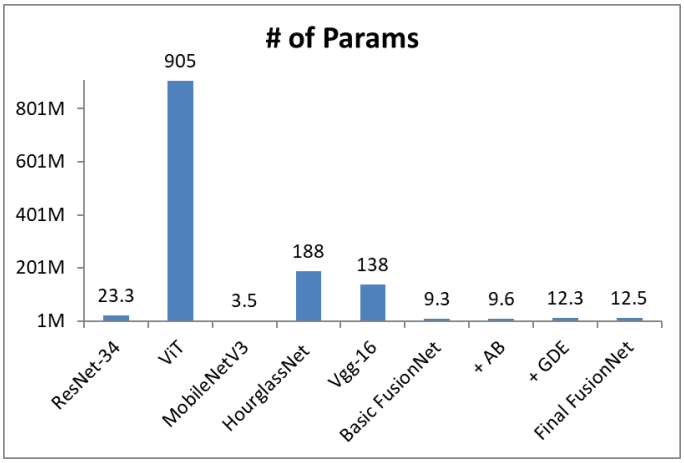


**Figure 1.** Comparison of numbers of model parameters. "+AB" and "+GDE" represent the basic FusionNet coupled with attention blocks and with GDE, respectively. FusionNet has about half the parameters of the ResNet model most commonly used in 6D object pose estimation studies, and is much lighter compared to other huge models.

## 2. Related Work

In the past, methods based on depth information dominated the field of 6D object pose estimation due to their accuracy and robustness [32,33]. However, it is not always possible to use depth information in outdoor or mobile environments. Therefore, methods of predicting 6D pose from only RGB images have received attention in recent years, although it is more challenging. In this section, we briefly review 6D object pose estimation methods using only RGB images. They are classified into two folds: CNN-based and Transformer-based methods.

### *2.1. CNN-Based Method*

Classical 6D object pose estimation methods can be categorized into indirect and direct methods. While direct methods directly regress object poses from input images, indirect methods predict robust intermediate representations, subsequently estimating object poses from the intermediate representations. For both categories, various CNN-based methods have been proposed [2,3,34–38]. In indirect methods, a popular intermediate representation is keypoint, achieving excellent performance in previous studies [2,39–42]. For example, Pavlakos et al. [40] localized a set of class-specific keypoints using a stacked hourglass CNN that outputs a pixel-wise heatmap for each keypoint. Oberweger et al. [38] trained an encoder–decoder network to predict heatmaps for the 2D projections of the corners of the object's 3D bounding box. However, since keypoint-based methods are usually vulnerable to occlusion, recent studies have been conducted for occlusion handling. Another common

intermediate representation is the coordinates of each image pixel in the 3D physical world, which provides a dense 2D–3D correspondence and is robust under occlusion [43–45]. For example, Haugaard et al. [45] obtained dense, continuous 2D–3D dense correspondence distributions of the object surface using a U-Net architecture. From 2D–3D keypoint or dense correspondences, the object pose can be computed using the PnP algorithm or predicted using the PnP model in an end-to-end manner. As more recent CNN-based methods, GDR-Net [46] directly regressed object poses from the intermediate geometric features regarding dense correspondences. EPro-PnP [20] translated the non-differentiable deterministic PnP operation into a differentiable probabilistic layer. With ResNet-34 as the backbone, both achieved state-of-the-art performance using only RGB images in 6D object pose estimation. One of the main concerns of the most recent CNN-based methods is estimating object poses without 3D CAD models, which are usually not available in real-world environments [47]. However, despite their impressive results in 6D object pose estimation, the problem of CNNs not being skilled at capturing long-range dependencies has been difficult to mitigate.

Most of CNN-based methods used VGG [48] or ResNet [21] as a backbone and few attempts have been made to redesign the backbone network.

### *2.2. Transformer-Based Method*

The remarkable success of Transformers in the NLP field has led to attempts to introduce Transformers into a variety of vision tasks. ViT first introduced Transformer to a classification task and applied the Transformer encoder to extract features. DETR [49] used the Transformer decoder to model object detection as an end-to-end dictionary lookup problem with learnable queries, successfully eliminating the need for manual processes such as non-maximum suppression. IPT [50] used a shared Transformer body with multi-heads and multi-tails for serving different low-level vision tasks such as deraining, denoising, and super-resolution. However, few attempts to use Transformers have been made in 6D object pose estimation. Recently, some studies showed that Transformers have a competitive performance in 6D object pose estimation as well [28–31,51,52]. PoET [28] is a Transformer-based framework that takes a single RGB image as input and estimates the 6D poses for all objects present in the image without object 3D CAD models. Bounding box information for each object is passed to the transformer decoder as an object query. Then, the output object queries are processed by a separate translation and rotation head. Video-Pose [52] is a Transformer-based framework that estimates accurate 6D object poses in videos. It leverages the temporal information from a video sequence for pose refinement. T6D-Direct [51] is a single-stage direct regression method with a transformer-based architecture built on DETR to perform 6D multi-object pose estimation. Trans6D [29] is a Transformer-based framework that predicts dense 2D–3D correspondence maps from an RGB input image. In order to further improve the performance of Trans6D, Trans6D+, a pure Transformer-based pose refinement module, was proposed that learns a transformation between the predicted pose and the ground-truth pose. YOLOPose [31] is a Transformer-based multi-object pose estimation framework based on keypoint regression. It jointly estimates bounding boxes, labels, translation parameters, and pixel coordinates of 3D keypoints for all objects in the given input image. CRT-6D [30] is a cascade of pose refinement Transformers for 6D object pose estimation. It iteratively updates an initial pose guess by performing self-attention over a sparse set of object keypoint features. Although some of them use CNNs to extract shallow features from input images, Transformer-based methods process features using only Transformers; thus, their performance is still limited, because Transformers have lower performance than CNNs of similar size when trained on small amounts of data.

## 3. FusionNet

### *3.1. Overall Architecture*

FusionNet is a hybrid model that combines the advantages of Transformer and CNN. Figure 2 provides an overview of the network architecture of the representative CNN

and Transformer models for vision tasks, ResNet-34 [21] and ViT [13], and the proposed FusionNet. ViT splits the input image into multiple patches, and the information within the patches is modeled by linear projection and position embedding. Then, the dependencies between the patches are modeled by self-attention operations in the subsequent Transformer encoder. ViT is a good model for vision tasks requiring global understanding of the input image/scene, such as classification, but not for vision tasks heavily depending on local features as well, such as object pose estimation. Furthermore, ViT directly uses high-dimensional input images in the Transformer encoder, which increases the computational complexity dramatically. To mitigate these limitations, FusionNet is basically designed to extract features using CNN blocks from input images. Specifically, referring to the design policy of ResNet-34 used in EPro-PnP [20], FusionNet generates feature maps at different scales through four stages in which several CNN blocks are stacked sequentially, while maintaining the same resolution as the input within each stage. Feature maps are downsampled using one $1 \times 1$ convolution with stride = 2 and batch normalization at the end of each stage. However, the CNN building block has a different structure from that of ResNet-34 and also has an attention block, which is described in detail in Section 3.2. At each stage, four, six, seven, and three CNN blocks are connected sequentially. In addition, the output of Stage-2 is fed into the GDE and concatenated with the output of Stage-3, where the long-range dependencies are captured. The GDE has the similar structure to the Transformer encoder of ViT, but it is much lighter because it uses low-dimensional feature maps as the input, which is elaborated in Section 3.3. The reason why GDE is placed in parallel with Stage-3 is as follows. As aforementioned, the convolutional operation encodes long-range dependencies by deeply stacking itself, and the dependencies cannot be sufficiently captured at early stages even if the attention block is included in the CNN blocks. However, early-stage feature maps require a huge amount of model parameters for GDE because of their large resolution. Therefore, we believe that placing the GDE in parallel with Stage-3, which is in the middle of the CNN pipeline, is more suitable in terms of model efficiency and functionality.

The full network architecture of FusionNet for 6D object pose estimation is shown in Figure 3. From an RGB input image, object areas are cropped and resized to $256 \times 256$ pixels. Object areas may be detected using existing detection methods, but this study assumes that they are given in advance. FusionNet extracts informative features using the CNN blocks and GDE from the resized object images and the features are fed into the EPro-PnP head [20] that consists of two sub-heads: one is a regression model for predicting translation parameters, the other is to extract a dense 3D coordinate map and a weight map via convolutional layers and to predict rotation parameters from the maps in an end-to-end manner using a PnP block that replaces the PnP algorithm. In the training phase, the translation head minimizes the L2 loss between predicted and ground-truth parameters, and the rotation head minimizes the KL divergence loss [20].

### *3.2. CNN Block*

As shown in Figure 2, the ResNet blocks of ResNet-34 consist of two sets of $3 \times 3$ convolution, batch normalization, and ReLU activation. In contrast, aiming to achieve good performance while ensuring that the network is lightweight, we extract features using only one $3 \times 3$ convolution and replace the other $3 \times 3$ convolution with two $1 \times 1$ convolutions. The $1 \times 1$ convolution reduces the channel size, reducing the number of model parameters (see Figure 1). Instead, we increase the number of output channels of convolution at each stage to 80, 160, 304, and 680, respectively. This modification also has the advantage of increasing the nonlinearity of the network, allowing for the network learning of more complex features. For this reason, we omit the ReLU activation layer.

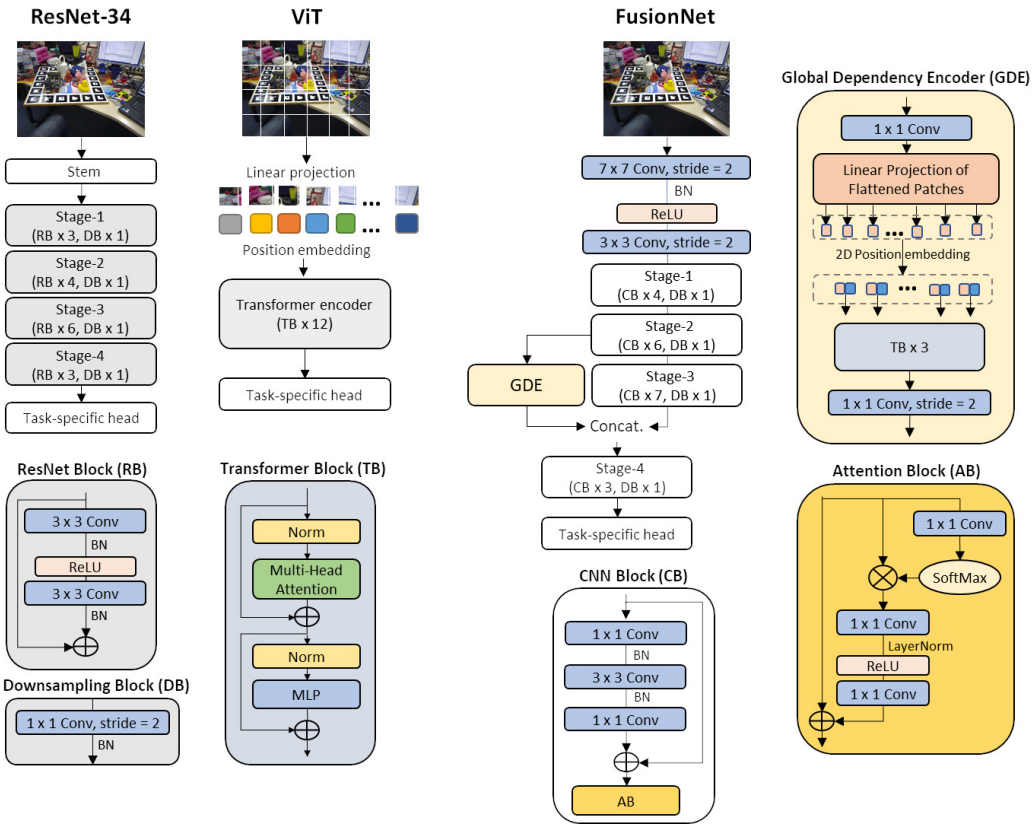


**Figure 2.** Network architecture of ResNet-34, ViT, and FusionNet. FusionNet partially includes the structures of ResNet-34 and ViT.

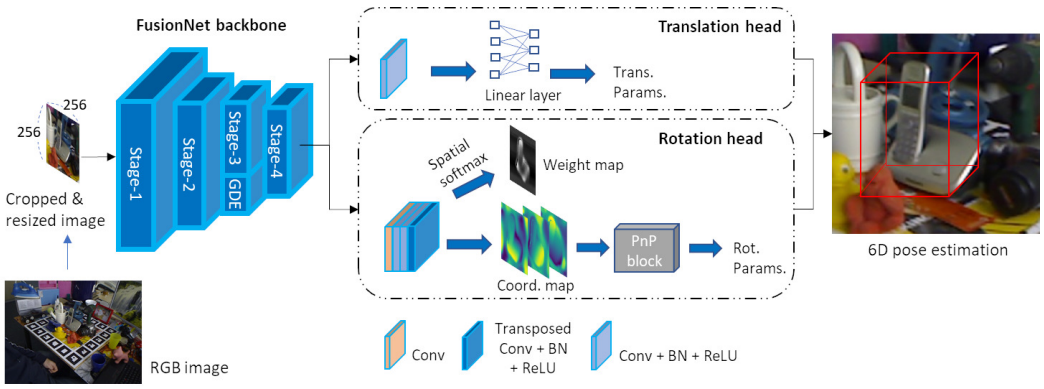


**Figure 3.** Process pipeline of FusionNet for 6D object pose estimation.

Inspired by [9,10,18], the attention mechanism is introduced to improve the ability of CNN blocks to capture the context information. A non-local block is connected to the second $1 \times 1$ convolution layer, in which non-local operations enhance features by aggregating information from other locations. For model simplicity, the simplified non-local block proposed in [10] is used. The simplified block is much lighter than the original non-local block, but with little decrease in accuracy.

To investigate how FusionNet facilitates feature learning, we visualize feature maps generated at each stage and the resulting coordinate maps in Figure 4. In the visualization, the regions of brighter colors correspond to stronger features. We can observe that our CNN blocks have similar capabilities to those of ResNet blocks in feature extraction. However, ResNet-34 has many bright regions, focusing on irrelevant objects. In contrast, FusionNet filters out unnecessary information using the attention block and more using GDE. This

advantage allows for FusionNet obtaining more reliable coordinate maps, enabling more accurate pose estimation in the PnP block.

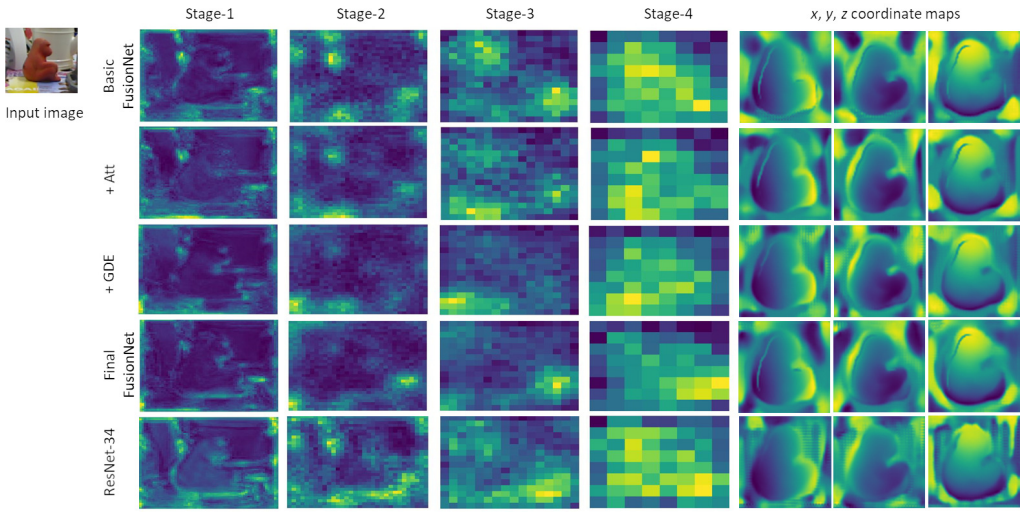


**Figure 4.** Visualization of feature maps generated at the end of each stage of ResNet-34 and FusionNet and the resulting coordinate maps.

### *3.3. GDE*

GDE has a similar structure to ViT's Transformer encoder, but unlike ViT, it has only three standard Transformer blocks. In addition, GDE does not directly process the input image. Instead, it receives the convolutional feature map from CNN blocks and splits it into multiple patches which are then reshaped into 1D tokens. Next, the tokens are linearly projected, summed with position embeddings, and fed to the first standard Transformer block. For position embedding, we use 2D sine position embedding because it may help to generalize better the pose estimation of objects of different scales [53]. GDE includes one $1 \times 1$ convolutional layer before the linear projection block and after the last Transformer block, respectively. The first one is to increase the number of channels in the input feature map, which allows us the capture of more features useful for pose estimation because each channel captures different features. The second one is to downsample the output dependency map by a factor of two to be concatenated with the feature map of Stage-3 as shown in Figure 2.

To summarize, we first extract features via multi-stage convolutions. Then, long-range dependencies between the features are captured by self-attention mechanism in the Transformer blocks of GDE. Finally, the dependency map is reshaped and downsampled, concatenated with the output of the Stage-3, and input to the Stage-4 of Fusion-Net.

The GDE structure has two advantages in terms of efficiency. First, it allows us a simple building of a hybrid model with minor modifications to the CNN-based body while taking full advantage of the Transformer's capabilities. Second, even if fusing two different architectures, it allows the light weight of the fusion model.

Unlike the attention block in the CNN blocks, GDE processes image blocks as sequence data at once, and explicitly encodes the dependencies of individual image blocks through the self-attention mechanism, which enhances the global understanding of feature maps. The attention block operates on a similar principle to GDE, but it calculates the contribution of individual convolutional localizations. This also explains why CNNs have a hard time mitigating the problem of weak long-range dependencies even when combined with attentions, whereas Transformer is better at capturing long-range dependencies.

## 4. Experimental Results and Discussion

### *4.1. Datasets and Metrics*

We used the LINEMOD dataset [54] consisting of 13 sequences for testing and training, each sequence containing approximately 1.2 K images annotated with 6D poses and 2D bounding boxes of a single object. A 3D CAD model of each object was also provided. The images were divided into training and test sets according to [43], with approximately 200 images per object for training. For data augmentation, we used the same synthetic data as in CDPN [55]. We used two common metrics for evaluation: ADD(-S) and 2D reprojection error. ADD measures whether the mean 3D distance between object's mesh vertices transformed by the ground-truth pose and by the predicted pose is less than a certain fraction of the object diameter. For example, ADD-0.1d determines that the predicted pose is correct when the distance is less than 10% of the object diameter and computes the percentage of images where the predicted pose is correct to all test images. A 2D reprojection error is the mean distance between the 2D projection of the object's 3D mesh vertices applying the predicted and the ground-truth pose, and the predicted pose is correct if the error is less than 5 pixels. For both metrics, we measure the percentage of images where the object pose is estimated correctly.

### *4.2. Experimental Setup*

For the convenience of implementation, FusionNet was implemented based on the open source code of EPro-PnP [20]. Our source code is accessible at https://github.com/helloyuning/FusionNet (accessed on 22 August 2023). For fair comparison, the general settings were the same as in EPro-PnP, except that we replaced the dense correspondence network with our FusionNet. The implementation was performed with PyTorch on a desktop computer (i7 2.5 GHz CPU and 32 GB RAM) with a single RTX 2060 GPU. For training, the RMSProp optimizer with $\alpha = 0.99$, $\epsilon = 1 \times 10^{-8}$, $\lambda = 0$, and $\mu = 0$ was used. The learning rate was set to $1 \times 10^{-4}$ and the number of epochs was 320. We adopted the fine-tuning strategy of EPro-PnP. FusionNet was first pre-trained on the ImageNet dataset [56] for image classification. The pre-trained model was then used as a backbone for CDPN and fine-tuned on the LINEMOD dataset for pose estimation. Finally, the fine-tuned model was combined with the EPro-PnP head for end-to-end pose estimation and fine-tuned again on the LINEMOD dataset for pose estimation.

Unfortunately, due to equipment limitations, it was not possible to use the entire ImageNet dataset to obtain the pre-trained model. Therefore, we only used 300 images for each of the 1000 categories of ImageNet dataset to obtain the pre-trained model. We also decreased the training batch size from 32 to 16. This is why the accuracy of CDPN and EPro-PnP is lower in our results, compared with those reported in the previous studies. Therefore, we focused on showing the superiority of FusionNet relative to CDPN and EPro-PnP in the same conditions. In our experiments, we found that the performance of the fine-tuning strategy used in EPro-PnP and FusionNet is heavily dependent on the accuracy of the pre-trained model. Therefore, in order not to be overly influenced by the pre-trained model for a fair comparison, we also trained all the models from scratch and compared their performance.

### *4.3. Ablation Study*

Compared to the baseline EPro-PnP model, FusionNet has three modifications: a newly designed CNN building block, introduction of an attention block within the CNN block, and introduction of GDE. Therefore, we analyzed how each modification contributes to the performance of FusionNet. Therefore, in the following tables, "EPro-PnP" represents the original EPro-PnP model with no modifications, "Basic FusionNet" represents the EPro-PnP model of which the ResNet blocks are replaced with the newly designed CNN blocks, and "Final FusionNet" represents the EPro-PnP model with all three modifications. In addition, "ViT-PnP" represents the ViT model of which the head was changed to the EPro-PnP header (Figure 3) for 6D object pose estimation. Table 1 shows the contribution

of each modification, where we trained the models from scratch for a fair comparison and to eliminate the effect of pre-training. The followings can be observed from the results:

- The original EPro-PnP model uses ResNet-34 as a backbone, and in our experiments, it seemed to rely heavily on model initialization. When trained from scratch, the ADD-0.1d score remained at 73.78.
- Using Transformers alone without the help of CNNs in the EPro-PnP framework significantly reduced accuracy. The ADD-0.1d score decreased by 27.16.
- Replacing the ResNet blocks in EPro-PnP with our CNN blocks significantly improved accuracy. The ADD-0.1d score increased to 77.63.
- The performance of Basic FusionNet was further improved when the attention block and the GDE were added. The ADD-0.1d score increased to 79.38 and 81.86, respectively. Finally, when both were included, the performance was most improved. The ADD-0.1d score increased up to 83.07.
- In GDE, we used 2D sine position embedding instead of 1D learnable position embedding used in ViT. This also contributed to improved performance, increasing the ADD-0.1d score by 0.69.

**Table 1.** Ablation study of FusionNet when trained from scratch. The values in parentheses indicate the degree of improvement achieved by each modification. The last row shows the results when using learnable position embedding [13] in GDE, instead of 2D sine position embedding.

| Model | ADD(-S) | | | Mean |
|---|---|---|---|---|
| | 0.02d | 0.05d | 0.1d | |
| EPro-PnP | 12.54 | 43.79 | 73.78 | 43.37 |
| ViT-PnP | 3.05 (−9.49) | 20.41 (−23.38) | 46.62 (−27.16) | 23.33 (−20.04) |
| Basic FusionNet | 15.61 (+3.07) | 49.82 (+6.03) | 77.63 (+3.85) | 47.69 (+4.32) |
| +AB | 17.23 (+4.69) | 51.78 (+7.99) | 79.38 (+5.60) | 49.46 (+6.09) |
| +GDE | 18.61 (+6.07) | 54.40 (+10.61) | 81.86 (+8.08) | 51.62 (+8.25) |
| Final FusionNet | 19.32 (+6.78) | 55.15 (+11.36) | 83.07 (+9.29) | 52.51 (+9.14) |
| w/o 2D embedding | 18.31 (+5.77) | 54.88 (+11.09) | 82.38 (+8.60) | 51.86 (+8.49) |

We then conducted the same ablation study on the trained models with pre-training. The results are shown in Table 2, and the followings can be observed:

- The performance of EPro-PnP was significantly improved with pre-training, e.g., from 73.78 to 92.61 at the ADD-0.1d score.
- Pre-training also contributed to FusionNet's performance improvement, but not as dependent as EPro-PnP. The ADD-0.1d score of Final FusionNet increased from 83.07 to 93.48.
- It is clear that the three modifications resulted in significant performance improvements over EPro-PnP.

Figure 5 shows the inference time of each model. Basic FusionNet was 1.5 times slower than EPro-PnP. The reasons are as follows: First, it has more CNN blocks in each stage than EPro-PnP (Figure 2); Second, its CNN building block has one more convolution layer than EPro-PnP's ResNet building block; Third, each convolution layer has more output channels than EPro-PnP. Final FusionNet has more processing blocks (AB and GDE take 3 ms and 4 ms, respectively); thus, is slower than Basic FusionNet. As a result, Final FusionNet requires approximately 2.67 times longer inference time than EPro-PnP. However, Final FusionNet is still fast (>62 fps) enough to operate in real time.

**Table 2.** Ablation study of FusionNet when trained with pre-training. The values in parentheses indicate the degree of improvement achieved by each modification.

| Model | ADD(-S) | | | Mean |
|---|---|---|---|---|
| | 0.02d | 0.05d | 0.1d | |
| EPro-PnP | 31.98 | 71.84 | 92.61 | 65.48 |
| Basic FusionNet | 33.09 (+1.11) | 72.62 (+0.78) | 92.89 (+0.28) | 66.20 (+0.72) |
| +AB | 34.75 (+2.77) | 74.60 (+2.76) | 93.01 (+0.40) | 67.45 (+1.97) |
| +GDE | 33.95 (+1.97) | 73.17 (+1.33) | 92.48 (−0.13) | 66.53 (+1.05) |
| Final FusionNet | 35.36 (+3.38) | 74.50 (+2.66) | 93.48 (+0.87) | 67.78 (+2.30) |

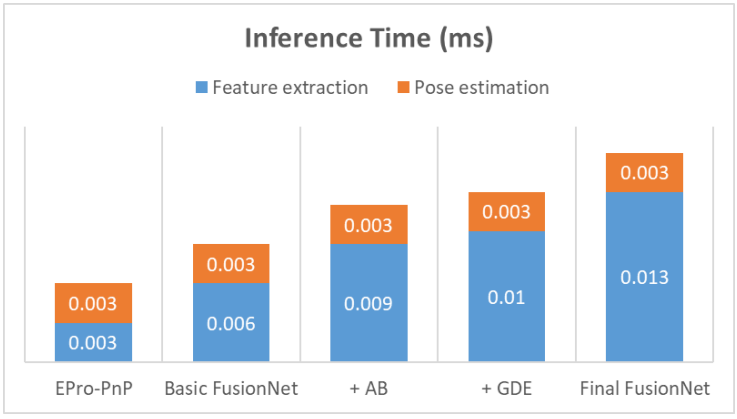


**Figure 5.** Model inference time. The pose estimation time is the time spent on the common head estimating rotation and translation parameters.

To ensure that the models were properly trained, the training errors over the number of epochs are visualized in Figure 6. In addition, to verify their generalization capabilities, the validation accuracies are visualized in Figure 7. For both EPro-PnP and FusionNet, the rate of decline decreased, but the training errors decreased steadily. The validation accuracies also steadily increased as the number of epochs increased. This means that the models were properly and sufficiently trained with good generalization capabilities. In terms of the training speed, FusionNet with the Transformer architecture was slower than EPro-PnP.

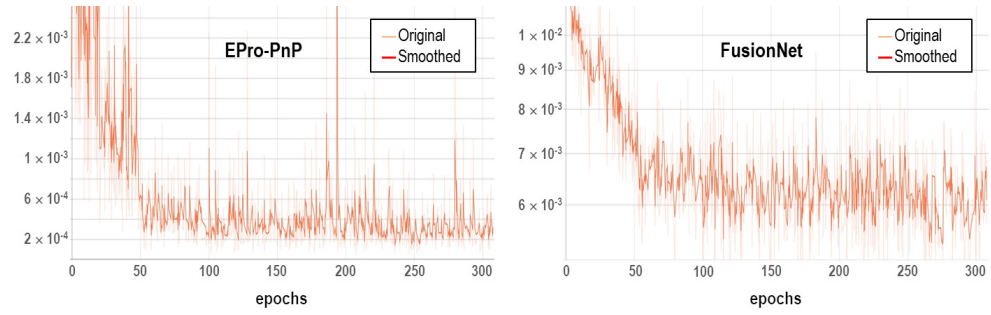


**Figure 6.** Training error/loss curves of EPro-PnP and FusionNet.

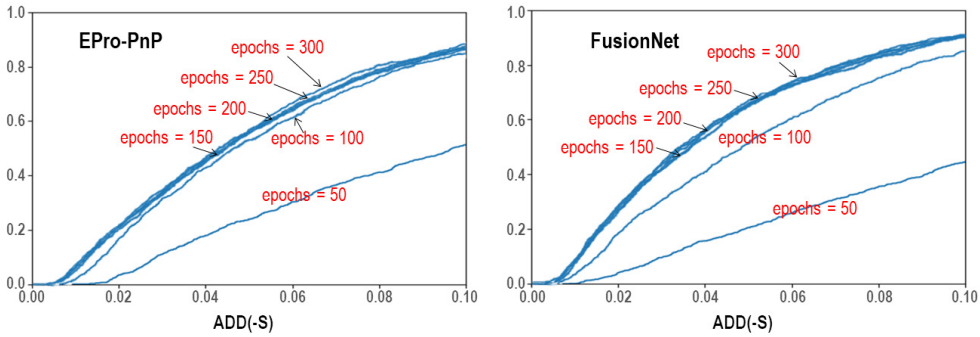


**Figure 7.** Validation accuracy curves of EPro-PnP and FusionNet.

Table 3 shows object-wise ADD-0.1d scores of FusionNet. Although object-dependent, each modification had a positive effect on most objects, and Final FusionNet, which included a combination of three modifications, received significantly higher ADD scores than EPro-PnP for all objects. The degree of improvement can be observed more clearly in Figure 8. The tendency has hardly changed with or without pre-training except for "Duck" and "Eggbox". However, we observed a difference in the overall dependence of each model on pre-training. For each object, the ADD scores significantly increased with pre-training. However, FusionNet had a weaker dependence than EPro-PnP, which is good for the practical use of the model. In fact, the pre-training is tedious and time consuming, particularly in our pre-training process where the model needs to be pre-trained twice using the ImageNet and LINEMOD datasets.

Table 4 shows 2D reprojection errors of FusionNet with and without modifications. Even the resulting values of the baseline model (EPro-PnP) are high (>90), so the degree of improvement does not seem significant, but all the modifications contributed to further increasing the values for most objects. The degree of improvement is clearly observed in Figure 9. The weaker dependence of FusionNet on pre-training was also observable. However, unlike the ADD scores, we found it difficult to interpret. For some objects, although each modification individually contributed to improving the performance of EPro-PnP, when all modifications were applied, the performance was rather lower than when some of the modifications were applied. This became more severe with pre-training. We believe that the reasons for this should be analyzed in depth, so we leave it for further research.

**Table 3.** Object-wise ADD-0.1d scores of FusionNet with different modifications. The values to the left and right of "/" represent the results when trained from scratch and when trained with pre-training, respectively.

| Object | EPro-PnP | Basic FusionNet | +AB | +GDE | Final FusionNet |
|---|---|---|---|---|---|
| Ape | 53.14/79.71 | 59.05/77.90 | 52.86/78.19 | 63.90/82.76 | 64.29/79.90 |
| Benchvise | 88.17/96.80 | 86.61/96.31 | 89.62/97.38 | 91.85/96.22 | 90.20/98.16 |
| Camera | 65.49/93.33 | 71.27/94.12 | 74.80/94.61 | 76.76/94.12 | 79.02/94.12 |
| Can | 75.10/96.95 | 79.43/97.15 | 81.30/98.03 | 82.19/97.54 | 84.25/97.24 |
| Cat | 58.38/89.22 | 63.07/89.32 | 71.46/88.72 | 78.04/91.32 | 74.65/90.72 |
| Driller | 78.39/94.25 | 82.66/96.04 | 87.71/95.74 | 87.12/95.84 | 86.92/96.63 |
| Duck | 60.28/80.09 | 56.81/79.44 | 51.08/79.34 | 61.60/70.14 | 66.67/78.40 |
| Eggbox | 97.56/99.72 | 98.78/99.72 | 99.34/99.81 | 99.15/99.53 | 99.25/99.62 |
| Glue | 80.12/97.59 | 86.00/98.17 | 91.31/97.49 | 81.18/97.10 | 89.67/98.65 |
| Holepuncher | 62.32/89.44 | 70.03/91.15 | 71.74/89.72 | 80.11/89.72 | 78.40/91.06 |
| Iron | 87.54/96.83 | 88.76/95.91 | 91.73/97.65 | 90.50/96.63 | 92.44/97.96 |
| Lamp | 88.87/99.14 | 94.63/98.94 | 93.86/99.42 | 97.31/99.04 | 96.74/99.33 |
| Phone | 63.83/90.84 | 72.05/93.39 | 75.07/93.01 | 74.41/92.26 | 77.43/93.39 |
| Mean | 73.78/92.61 | 77.63/92.89 | 79.38/93.01 | 81.86/92.48 | 83.07/93.48 |

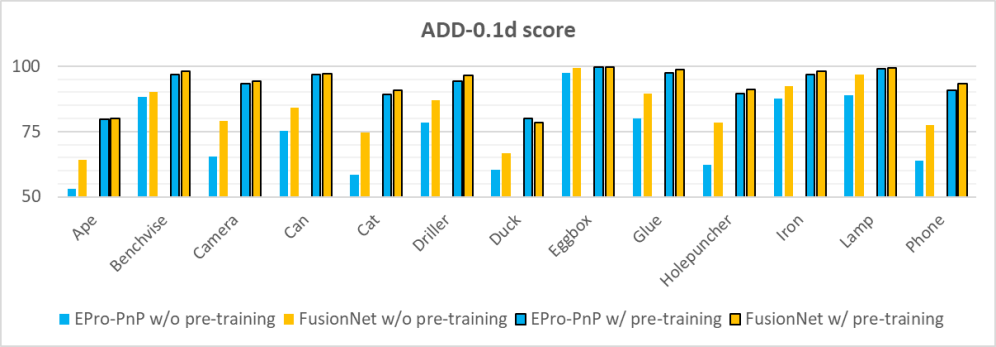


**Figure 8.** Comparison of ADD-0.1d scores between EPro-PnP and FusionNet.

**Table 4.** Object-wise 2D reprojection errors of FusionNet with different modifications. The values to the left and right of "/" represent the results when trained from scratch and when trained with pre-training, respectively.

| Object | EPro-PnP | Basic FusionNet | +AB | +GDE | Final FusionNet |
|---|---|---|---|---|---|
| Ape | 97.52/98.38 | 97.81/98.76 | 98.10/98.67 | 98.10/98.95 | 98.00/98.48 |
| Benchvise | 96.70/98.64 | 95.34/98.45 | 96.51/98.74 | 96.41/99.13 | 95.64/99.03 |
| Camera | 95.10/98.82 | 97.35/99.22 | 97.35/99.41 | 97.45/99.22 | 98.43/99.02 |
| Can | 93.11/99.31 | 94.98/99.21 | 96.46/99.41 | 96.16/99.11 | 96.75/99.61 |
| Cat | 98.20/99.60 | 98.80/99.50 | 98.90/99.60 | 99.00/99.30 | 99.20/99.80 |
| Driller | 90.39/96.53 | 90.68/96.23 | 92.67/98.02 | 93.46/97.62 | 94.65/98.22 |
| Duck | 98.40/98.87 | 98.40/98.87 | 97.93/98.97 | 98.40/99.06 | 98.22/98.78 |
| Eggbox | 98.59/98.97 | 98.78/99.15 | 99.06/99.15 | 98.59/99.44 | 99.06/99.06 |
| Glue | 93.63/99.23 | 97.01/99.03 | 97.78/99.13 | 95.46/98.65 | 97.88/99.13 |
| Holepuncher | 98.86/99.81 | 98.95/99.81 | 99.14/99.81 | 98.86/100 | 98.95/99.62 |
| Iron | 91.52/96.73 | 93.97/97.14 | 93.46/97.65 | 93.56/97.14 | 94.59/97.14 |
| Lamp | 90.88/97.31 | 94.15/96.83 | 94.53/98.08 | 94.53/97.31 | 94.82/97.79 |
| Phone | 92.26/98.96 | 94.90/98.39 | 95.28/99.15 | 96.32/98.58 | 96.32/98.96 |
| Mean | 95.01/98.55 | 96.24/98.51 | 96.71/98.91 | 96.64/98.73 | 97.12/98.82 |

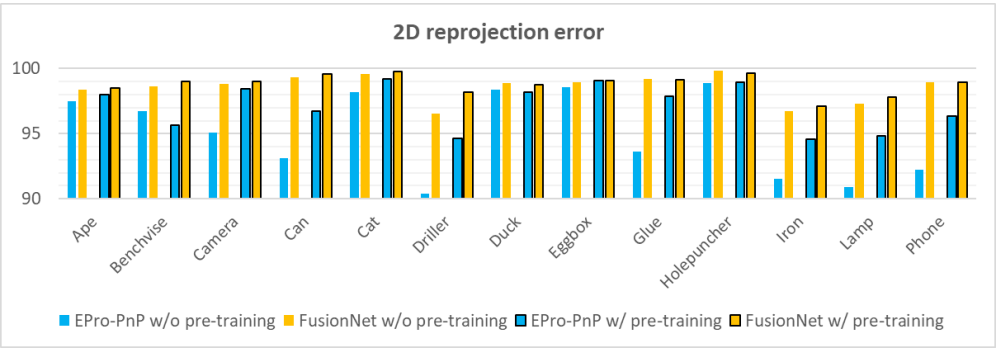


**Figure 9.** Comparison of 2D reprojection errors between EPro-PnP and FusionNet.

### *4.4. Comparison with Other 6D Object Pose Estimation Methods*

In Table 5, the results of CDPN, EPro-PnP, and FusionNet were obtained in our experiments and the others were brought from the related papers. Although CDPN and GDR-Net used additional synthetic image datasets for training, all the results were obtained using the same publicly available train and test datasets. All the methods performed pre-training using the ImageNet dataset, but for CDPN, EPro-PnP, and FusionNet, the entire ImageNet dataset was not used.

As mentioned earlier, we could not obtain pre-trained models using the entire ImageNet dataset due to equipment limitations. However, as shown in Table 5, FusionNet achieved comparable accuracy to that of methods (such as HybridPose and GDR-Net) with pre-training using the entire ImageNet dataset, despite pre-training using only a small portion of the ImageNet dataset. This demonstrates the great potential of FusionNet, and it is expected that pre-training FusionNet using the entire ImageNet dataset will result in higher accuracy. The accuracy of GDR-Net may be due, in part, to the use of additional synthetic training datasets. Considering these points, we can say that FusionNet outperforms other methods without pose refinement. Under the same conditions, FusionNet achieved higher accuracy than EPro-PnP, which provides state-of-the-art performance using only RGB images without pose refinement. DPOD and PVNet+RePOSE have shown that pose refinement can significantly increase accuracy. However, their accuracies were much lower than that of FusionNet without pose refinement. Since PVNet focused on occlusion handling, it seemed to be less accurate without the refinement by RePOSE on the LINEMOD dataset with no or mild occlusion.

Table 6 shows object-wise ADD-0.1d scores of 6D object pose estimation methods. The GDR-Net does not provide the object-wise scores and DPOD and PVNet+RePOSE do not provide the object-wise scores without pose refinement. FusionNet achieved object-dependent but comparable accuracy to that of DPOD and PVNet+RePOSE, without pose refinement. Except for "Duck" and "Eggbox", FusionNet consistently achieved better accuracy than EPro-PnP.

Figure 1 shows the number of model parameters of FusionNet and the backbone networks commonly used in vision tasks. We can see that FusionNet is very lightweight despite the state-of-the-art performance shown in the previously presented results. Compared to ResNet-34, which is the most popular backbone, FusionNet has nearly 2× fewer model parameters even after fusing the Transformer (GDE). This efficiency stems primarily from our CNN blocks replacing ResNet blocks. In addition, despite significant contributions to FusionNet's performance improvement, the attention block and GDE slightly increased the model parameters. Compared to ViT, FusionNet has nearly 75× fewer model parameters, which demonstrates the efficiency of FusionNet for practical use under real-world conditions.

**Table 5.** ADD score comparison with other 6D object pose estimation methods. Some methods include pose refinement processes.

| **Method** | **w/o Refinement** | | | **w/ Refinement** |
|---|---|---|---|---|
| | **0.02 d** | **0.05 d** | **0.1 d** | **0.1 d** |
| CDPN [55] | 16.17 | 54.20 | 81.87 | - |
| HybridPose [37] | - | - | 91.30 | - |
| GDR-Net [46] | 35.60 | 76.00 | 93.70 | - |
| DPOD [57] | - | - | 82.98 | 95.15 |
| PVNet+RePOSE [58] | - | - | 86.93 | 96.10 |
| EPro-PnP [20] | 31.98 | 71.84 | 92.61 | - |
| Final FusionNet | 35.36 | 74.50 | 93.48 | - |

**Table 6.** Object-wise ADD-0.1d scores of 6D object pose estimation methods.

| Object | CDPN | HybridPose | GDR-Net | DPOD | PVNet+RePose | EPro-PnP | Final FusionNet |
|---|---|---|---|---|---|---|---|
| Ape | 56.76 | 63.10 | - | 87.73 | 79.50 | 79.71 | 79.90 |
| Benchvise | 91.56 | 99.90 | - | 98.45 | 100 | 96.80 | 98.16 |
| Camera | 83.14 | 90.40 | - | 96.07 | 99.20 | 93.33 | 94.12 |
| Can | 88.29 | 98.50 | - | 99.71 | 99.80 | 96.95 | 97.24 |
| Cat | 70.86 | 89.40 | - | 94.71 | 97.90 | 89.22 | 90.72 |
| Driller | 88.11 | 98.50 | - | 98.80 | 99.00 | 94.25 | 96.63 |
| Duck | 58.40 | 65.00 | - | 86.29 | 80.30 | 80.09 | 78.40 |
| Eggbox | 98.87 | 100 | - | 99.91 | 100 | 99.72 | 99.62 |
| Glue | 96.81 | 98.80 | - | 96.82 | 98.30 | 97.59 | 98.65 |
| Holepuncher | 67.94 | 89.70 | - | 86.87 | 96.90 | 89.44 | 91.06 |
| Iron | 91.83 | 100 | - | 100 | 100 | 96.83 | 97.96 |
| Lamp | 94.72 | 99.50 | - | 96.84 | 99.80 | 99.14 | 99.33 |
| Phone | 76.96 | 94.90 | - | 94.69 | 98.90 | 90.84 | 93.39 |
| Mean | 81.87 | 91.30 | 93.70 | 95.15 | 96.10 | 92.61 | 93.48 |

### *4.5. Discussion and Limitations*

There are a few things that need to be discussed and confirmed. First, existing PnP-based 6D object pose estimation methods have continuously improved feature extraction capabilities using CNNs of various architectures, but there is still room for improvement. This is the main motivation of this study. From the results presented above, we confirmed that FusionNet achieved more accurate pose estimation by improving the feature extraction capability of EPro-PnP. Second, the strategic introduction of Transformers into convolutional structures was expected to facilitate global features extraction, which plays a critical role in object pose estimation. From the experimental results, we confirmed that the GDE block of Transformer structure contributed to FusionNet achieving higher accuracy in pose estimation. Third, this study assumed that object areas are accurately detected in advance. However, object areas are often detected incorrectly due to incompleteness of object detection methods, occlusion, small size, and so on. Incorrect detection of object areas negatively affects FusionNet's performance, but it was not considered in this study. Fourth, in this study, the GDE block was simply designed using only three standard Transformer blocks for the model efficiency of FusionNet. However, for higher accuracy in pose estimation, it can be designed by stacking more Transformers with advanced architectures. The design optimization of GDE is beyond our scope and was not considered in this study.

Experimental results on the performance of FusionNet show that FusionNet has some limitations. FusionNet's performance is still highly dependent on tedious and time-consuming pre-training. With the introduction of Transformer, training time has increased significantly, approximately doubling the training time of EProPnP. FusionNet is lighter than ResNet-based models, including EPro-PnP, but its longer inference time can be another obstacle to the practical use of FusionNet.

## 5. Conclusions and Future Work

In this paper, we proposed FusionNet, which is a mixture of CNN and Transformer, to take advantages of both architectures. The newly designed CNN blocks with attention mechanism enabled FusionNet to efficiently extract informative features for 6D object pose estimation. The newly designed Transformer, GDE, helped FusionNet explicitly capture the long-range dependencies. As a result, FusionNet achieved high accuracy in 6D object pose estimation on the LINEMOD dataset while significantly reducing the model parameters. It outperformed other 6D object pose estimation methods, including EPro-PnP, which achieved state-of-the-art performance in 6D object pose estimation using RGB images.

However, as mentioned in Section 4.3, FusionNet behaved unexpectedly for some objects, when all the proposed modifications were applied. We need to analyze the results in depth and find ways to further improve FusionNet's performance through future study.

In addition, as mentioned in Section 1, we stated that FusionNet can be used in other vision tasks by changing the heads; thus, applying FusionNet to other vision tasks and analyzing its performance would be an interesting future study topic.

**Author Contributions:** Conceptualization, Y.Y. and H.P.; Funding acquisition, H.P.; Methodology, Y.Y. and H.P.; Software, Y.Y.; Supervision, H.P.; Validation, Y.Y. and H.P.; Writing—original draft, Y.Y.; Writing—review and editing, H.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Research Foundation of Korea (NRF) Grant by the Korean Government through the MSIT under Grant 2021R1F1A1045749.

**Data Availability Statement:** The data that support the findings of this study are publicly available in the online repository: https://bop.felk.cvut.cz/datasets/ (accessed on 22 August 2023).

**Conflicts of Interest:** We have no conflict of interest to declare.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| PnP | Perspective-n-Point |
| CNN | Convolutional Neural Network |
| NLP | Natural Language Processing |
| ViT | Vision Transformer |
| GDE | Global Dependency Encoder |
| AB | Attention Block |

## References

1. Lepetit, V.; Moreno-Noguer, F.; Fua, P. EPnP: An Accurate O(n) Solution to the PnP Problem. *Int. J. Comput. Vis.* **2009**, *81*, 155–166. [CrossRef]
2. 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 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 3848–3856. [CrossRef]
3. Tekin, B.; Sinha, S.; Fua, P. Real-Time Seamless Single Shot 6D Object Pose Prediction. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 292–301. [CrossRef]
4. Hu, Y.; Hugonot, J.; Fua, P.; Salzmann, M. Segmentation-Driven 6D Object Pose Estimation. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 3380–3389. [CrossRef]
5. Xie, S.; Girshick, R.; Dollar, P.; Tu, Z.; He, K. Aggregated Residual Transformations for Deep Neural Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5987–5995. [CrossRef]
6. Bhatt, D.; Patel, C.; Talsania, H.; Patel, J.; Vaghela, R.; Pandya, S.; Modi, K.; Ghayvat, H. CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope. *Electronics* **2021**, *10*, 2470. [CrossRef]
7. Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; Wei, Y. Relation Networks for Object Detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 3588–3597. [CrossRef]
8. Zhang, H.; Dana, K.; Shi, J.; Zhang, Z.; Wang, X.; Tyagi, A.; Agrawal, A. Context Encoding for Semantic Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7151–7160. [CrossRef]
9. Wang, X.; Girshick, R.; Gupta, A.; He, K. Non-local Neural Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 7794–7803. [CrossRef]
10. Cao, Y.; Xu, J.; Lin, S.; Wei, F.; Hu, H. GCNet: Non-Local Networks Meet Squeeze-Excitation Networks and Beyond. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Republic of Korea, 27–28 October 2019; pp. 1971–1980. [CrossRef]
11. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.u.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 5998–6008.

12. Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; Bowman, S. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Proceedings of the 7th International Conference on Learning Representations (ICLR), New Orleans, LA, USA, 6–9 May 2019.
13. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**, arXiv:2010.11929.
14. Chu, X.; Tian, Z.; Zhang, B.; Wang, X.; Shen, C. Conditional Positional Encodings for Vision Transformers. *arXiv* **2023**, arXiv:2102.10882.
15. Han, K.; Xiao, A.; Wu, E.; Guo, J.; Xu, C.; Wang, Y. Transformer in Transformer. *arXiv* **2021**, arXiv:2103.00112.
16. LeCun, Y.; Haffner, P.; Bottou, L.; Bengio, Y. Object Recognition with Gradient-Based Learning. In *Shape, Contour and Grouping in Computer Vision*; Springer: Berlin/Heidelberg, Germany, 1999; pp. 319–345. [CrossRef]
17. Wu, H.; Xiao, B.; Codella, N.; Liu, M.; Dai, X.; Yuan, L.; Zhang, L. CvT: Introducing Convolutions to Vision Transformers. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 22–31. [CrossRef]
18. Xu, L.; Guan, Y.; Jin, S.; Liu, W.; Qian, C.; Luo, P.; Ouyang, W.; Wang, X. ViPNAS: Efficient Video Pose Estimation via Neural Architecture Search. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 16067–16076. [CrossRef]
19. Xu, Y.; Zhang, J.; Zhang, Q.; Tao, D. ViTPose: Simple Vision Transformer Baselines for Human Pose Estimation. *arXiv* **2022**, arXiv:2204.12484.
20. Chen, H.; Wang, P.; Wang, F.; Tian, W.; Xiong, L.; Li, H. EPro-PnP: Generalized End-to-End Probabilistic Perspective-n-Points for Monocular Object Pose Estimation. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 2771–2780. [CrossRef]
21. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
22. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520. [CrossRef]
23. Guo, J.; Han, K.; Wu, H.; Tang, Y.; Chen, X.; Wang, Y.; Xu, C. CMT: Convolutional Neural Networks Meet Vision Transformers. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 12165–12175. [CrossRef]
24. Liang, J.; Cao, J.; Sun, G.; Zhang, K.; Gool, L.V.; Timofte, R. SwinIR: Image Restoration Using Swin Transformer. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Montreal, BC, Canada, 11–17 October 2021; pp. 1833–1844. [CrossRef]
25. Li, X.; Xiang, Y.; Li, S. Combining convolutional and vision transformer structures for sheep face recognition. *Comput. Electron. Agric.* **2023**, *205*, 107651. .: 10.1016/j.compag.2023.107651. [CrossRef]
26. He, L.; He, L.; Peng, L. CFormerFaceNet: Efficient Lightweight Network Merging a CNN and Transformer for Face Recognition. *Appl. Sci.* **2023**, *13*, 6506. [CrossRef]
27. Mogan, J.N.; Lee, C.P.; Lim, K.M.; Ali, M.; Alqahtani, A. Gait-CNN-ViT: Multi-Model Gait Recognition with Convolutional Neural Networks and Vision Transformer. *Sensors* **2023**, *23*, 3809. [CrossRef] [PubMed]
28. Jantos, T.; Hamdad, M.A.; Granig, W.; Weiss, S.; Steinbrener, J. PoET: Pose Estimation Transformer for Single-View, Multi-Object 6D Pose Estimation. In Proceedings of the Conference on Robot Learning, Auckland, New Zealand, 14–18 December 2022.
29. Zhang, Z.; Chen, W.; Zheng, L.; Leonardis, A.; Chang, H.J. Trans6D: Transformer-Based 6D Object Pose Estimation and Refinement. In Proceedings of the Computer Vision–ECCV 2022 Workshops, Tel Aviv, Israel, 23–27 October 2022; pp. 112–128. [CrossRef]
30. Castro, P.; Kim, T. CRT-6D: Fast 6D Object Pose Estimation with Cascaded Refinement Transformers. In Proceedings of the 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2–7 January 2023; pp. 5735–5744. [CrossRef]
31. Periyasamy, A.S.; Amini, A.; Tsaturyan, V.; Behnke, S. YOLOPose V2: Understanding and improving transformer-based 6D pose estimation. *Robot. Auton. Syst.* **2023**, *168*, 104490. .: 10.1016/j.robot.2023.104490. [CrossRef]
32. 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 (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3583–3592. [CrossRef]
33. Hinterstoisser, S.; Lepetit, V.; Rajkumar, N.; Konolige, K. Going Further with Point Pair Features. In Proceedings of the 14th European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer: Cham, Switzerland, 2016; pp. 834–848.
34. Kendall, A.; Grimes, M.; Cipolla, R. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2938–2946. [CrossRef]
35. Xiang, Y.; Schmidt, T.; Narayanan, V.; Fox, D. PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *arXiv* **2018**, arXiv:1711.00199.

36. 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.
37. Song, C.; Song, J.; Huang, Q. HybridPose: 6D Object Pose Estimation Under Hybrid Representations. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 428–437. [CrossRef]
38. Oberweger, M.; Rad, M.; Lepetit, V. Making Deep Heatmaps Robust to Partial Occlusions for 3D Object Pose Estimation. In Proceedings of the 15th European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018 ; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Springer: Cham, Switzerland, 2018; pp. 125–141.
39. Peng, S.; Zhou, X.; Liu, Y.; Lin, H.; Huang, Q.; Bao, H. PVNet: Pixel-Wise Voting Network for 6DoF Object Pose Estimation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 3212–3223. [CrossRef] [PubMed]
40. Pavlakos, G.; Zhou, X.; Chan, A.; Derpanis, K.G.; Daniilidis, K. 6-DoF object pose from semantic keypoints. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 2011–2018. [CrossRef]
41. Zhao, Z.; Peng, G.; Wang, H.; Fang, H.S.; Li, C.; Lu, C. Estimating 6D Pose From Localizing Designated Surface Keypoints. *arXiv* **2018**, arXiv:1812.01387.
42. Ullah, F.; Wei, W.; Daradkeh, Y.I.; Javed, M.; Rabbi, I.; Al Juaid, H.; Ali, S. A Robust Convolutional Neural Network for 6D Object Pose Estimation from RGB Image with Distance Regularization Voting Loss. *Sci. Program.* **2022**, *2022*, 2037141. [CrossRef]
43. Brachmann, E.; Michel, F.; Krull, A.; Yang, M.Y.; Gumhold, S.; Rother, C. Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3364–3372. [CrossRef]
44. Park, K.; Patten, T.; Vincze, M. Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 7667–7676. [CrossRef]
45. Haugaard, R.; Buch, A. SurfEmb: Dense and Continuous Correspondence Distributions for Object Pose Estimation with Learnt Surface Embeddings. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 6739–6748. [CrossRef]
46. Wang, G.; Manhardt, F.; Tombari, F.; Ji, X. GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 16606–16616. [CrossRef]
47. Liu, Y.; Wen, Y.; Peng, S.; Lin, C.; Long, X.; Komura, T.; Wang, W. Gen6D: Generalizable Model-Free 6-DoF Object Pose Estimation from RGB Images. In Proceedings of the 17th European Conference on Computer Vision (ECCV), Tel Aviv, Israel, 23–27 October 2022 ; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Springer: Cham, Switzerland, 2022; pp. 298–315.
48. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015; Bengio, Y., LeCun, Y., Eds.
49. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the 16th European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020 ; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., Eds.; Springer: Cham, Switzerland, 2020; pp. 213–229.
50. Chen, H.; Wang, Y.; Guo, T.; Xu, C.; Deng, Y.; Liu, Z.; Ma, S.; Xu, C.; Xu, C.; Gao, W. Pre-Trained Image Processing Transformer. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 12294–12305. [CrossRef]
51. Amini, A.; Periyasamy, A.S.; Behnke, S. T6D-Direct: Transformers For Multi-Object 6D Pose Direct Regression. In Proceedings of the 43rd DAGM German Conference on Pattern Recognition, Bonn, Germany, 28 September–1 October 2021; pp. 530–544. [CrossRef]
52. Beedu, A.; Alamri, H.; Essa, I. Video based Object 6D Pose Estimation using Transformers. In Proceedings of the NeurIPS 2022 Workshop on Vision Transformers: Theory and Applications, New Orleans, LA, USA, 28 November–9 December 2022.
53. Yang, S.; Quan, Z.; Nie, M.; Yang, W. TransPose: Keypoint Localization via Transformer. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 11782–11792. [CrossRef]
54. Hinterstoisser, S.; Cagniart, C.; Ilic, S.; Sturm, P.; Navab, N.; Fua, P.; Lepetit, V. Gradient Response Maps for Real-Time Detection of Textureless Objects. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 876–888. [CrossRef] [PubMed]
55. Li, Z.; Wang, G.; Ji, X. CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 7677–7686. [CrossRef]
56. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255. [CrossRef]

57. Zakharov, S.; Shugurov, I.; Ilic, S. DPOD: 6D Pose Object Detector and Refiner. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1941–1950. [CrossRef]
58. Iwase, S.; Liu, X.; Khirodkar, R.; Yokota, R.; Kitani, K.M. RePOSE: Fast 6D Object Pose Refinement via Deep Texture Rendering. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 3283–3292. [CrossRef]