Expeditious Object Pose Estimation for Autonomous Robotic Grasping

Sri Aditya Deevi[0000−0002−1463−8326] and Deepak Mishra

Indian Institute of Space Science and Technology
dsriaditya999@gmail.com, deepak.mishra@iist.ac.in

Abstract. The ability of a robot to sense and "perceive" its surroundings to interact and influence various objects of interest by grasping them, using vision-based sensors is the main principle behind vision based Autonomous Robotic Grasping. To realise this task of autonomous object grasping, one of the critical sub-tasks is the 6D Pose Estimation of a known object of interest from sensory data in a given environment. The sensory data can include RGB images and data from depth sensors, but determining the object's pose using only a single RGB image is cost-effective and highly desirable in many applications. In this work, we develop a series of convolutional neural network-based pose estimation models without post-refinement stages, designed to achieve high accuracy on relevant metrics for efficiently estimating the 6D pose of an object, using only a single RGB image. The designed models are incorporated into an end-to-end pose estimation pipeline based on Unity and ROS Noetic, where a UR3 Robotic Arm is deployed in a simulated pick-and-place task. The pose estimation performance of the different models is compared and analysed in both *same-environment* and *cross-environment* cases utilising synthetic RGB data collected from cluttered and simple simulation scenes constructed in Unity Environment. In addition, the developed models achieved high *Average Distance* (ADD) metric scores greater than 93% for most of the real-life objects tested in the LINEMOD dataset and can be integrated seamlessly with any robotic arm for estimating 6D pose from only RGB data, making our method effective, efficient and generic.

Keywords: 6D Pose Estimation · Autonomous Robotic Grasping · Deep Learning · Convolutional Neural Networks (CNNs)

1 Introduction

Autonomous Robotic Grasping is the ability of an "intelligent" robot to perceive its immediate environment and *grasp* the objects under consideration. This fundamental ability to grasp object can prove to be invaluable in various applications across a variety of domains. For example, industrial robots can be used for assisting human professionals in performing versatile and repetitive processing tasks such as pick-and-place, assembly and packaging whereas domestic robots can provide support to elderly or disabled people for their day to day grasping tasks.

6D Object Pose Estimation is a critical aspect that helps the robot to get aware of the target object to enable successful grasping. Based on the overall methodology, the object pose estimation approaches can be classified into : Correspondence-based [2][13], Template-based [12][16] and Voting-based [8][10] methods. Note that, each of these methods can be classified further depending upon whether depth information along with RGB data (RGB-D) is used or not.

In this work, a series of convolutional neural network-based pose estimation models without post-refinement stages are designed to efficiently and effectively estimate the 6D pose of an object, using only a single RGB image. The developed models were also benchmarked on various real-life objects present in the LINEMOD dataset. This paper is organized into the following sections : Section 2 is a basic overview that helps in establishing a high-level understanding of the entire pose-estimation pipeline. Various details related to pose estimation models designed are provided as a part of Section 3. Section 4 specifies the training and testing configuration utilized for evaluating these pose estimation models. Section 5 includes various quantitative and graphical results along with inferences for the collected Unity Synthetic dataset and on a popular object pose estimation dataset LINEMOD. Section 6 concludes the paper by reviewing some avenues of further research.

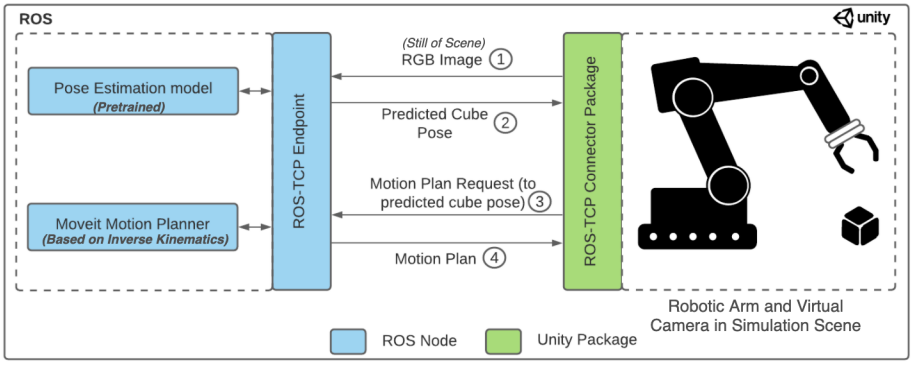

Fig. 1: Bird's Eye View of Pose Estimation Pipeline for object Pick and Place.The numbers ①, ②, ③ and ④ indicate the chronological sequence of events that take place during test phase. Ideally, the object of interest is picked up from the initial location after estimating its pose and placed in its target position, after planning the trajectory for the motion.

2 Overview of the Pose Estimation Pipeline

To create an end-to-end pose estimation pipeline for performing a pick and place task Unity Editor for Robotic Simulation is utilized. A high level flow diagram describing the necessary sequence of events that take place for the picking the object and placing it at the target location is shown in Fig. 1.

2.1 Phases of the Approach

The whole approach can be broadly divided into two main phases :

- *Train Phase* – In this phase, the pose estimation model is *trained* to predict the pose of the object of interest. This phase includes various subtasks including setting up the robotic arm and virtual camera in the simulation scene, configuring domain randomizers (see Section 2.2), data collection and training the model.
- *Test Phase* – In this phase, the pose estimation model is deployed and integrated into the simulated pick and place task. This phase uses the pretrained model from the Train Phase to predict the object pose which would be given to the MoveIt Motion Planning service for trajectory generation. Then using Unity *Articulation Bodies*, the arm moves according to the calculated trajectory to pick and place the object on a target mat.

2.2 Synthetic Data Collection

One of the important subtasks in the Train Phase is to collect data from the simulation scenario for training, validation and testing the models. Data includes RGB images (Here, resolution = 650×400 px) from the Virtual Camera (equipped with Unity's Perception Computer Vision Package) and *capture* files containing the ground truth annotation information. Note that, along with ground truth annotation pose labels, data regarding the sensor (in this case, a virtual camera) such as camera intrinsic matrix, the pose shift of the sensor w.r.t World frame, and sensor ID are also recorded.

Fig. 2: A collection of images depicting the superimposed effect of all domain randomizers applied to the simulation scenario (specifically, here *Cluttered Scene* is considered). The object under consideration (*cube*) is highlighted (green 3D bounding box)

Domain Randomization [12] a simple technique for bridging the simulation-reality gap for synthetic data, where instead of collecting data and training a model on a single simulated environment, we randomize the simulator to expose the model to a wide range of environments at training time. In this work, this idea is used extensively (See Fig. 2), and the different custom types of randomizers are configured for randomizing pose of different objects and camera (w.r.t World Frame) in the scene and also for randomizing the colour, intensity and direction of light.

3 Pose Estimation Models

Deep Learning based models are mainly utilized in this work, for single-object (of interest), single-instance 6D pose estimation to enable tasks such as pick and place. The design of the models was carried out iteratively keeping in mind, the following desired qualities :

- *Efficiency* – Use of only RGB image and no depth information
- *Speed* – Use of no post hoc refinement stages for pose prediction
- *Accuracy* – Good performance on relevant metrics such as Average Distance (ADD) of 3D model points

Each model acts as a baseline for demonstrating the performance improvement of the next model, which is designed to address the shortcomings of the previous model.

3.1 Model-1 : UnityVGG16

The overall architecture of Model-1 *UnityVGG16* is shown in Fig. 3.

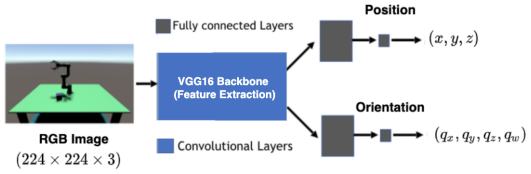

Fig. 3: Model Architecture of *UnityVGG16*. It is a template-based (implicit) approach that directly regresses the pose information from the RGB image.

The image features are extracted by the VGG16 [9] backbone.The network consists two heads made up of fully-connected layers, for regressing the 3D position (x, y, z) and orientation quaternion (q_x, q_y, q_z, q_w) of the object of interest, by the utilizing the extracted features.

3.2 Model-2 : Pose6DSSD

In order to improve the performance of pose estimation, the second model designed is *Pose6DSSD*[3]. The complete architecture of the model is illustrated in Fig. 4. Some of the ideas for designing the model were taken from [11].

As it is a correspondence based approach, we first the regress the 2D image coordinates of certain kepoints, which in this approach, are the 8 corners and the centroid of the 3D bounding box around the object of interest. The main

[3] Stands for 6D Pose **S**ingle **S**tage **D**etector

feature extraction backbone consists of 27 convolutional layers with residual skip links, and has been adapted from ResNet34 architecture[3]. There are no fully connected layers used as opposed to Model-1 *UnityVGG16*, to limit the number of parameters.

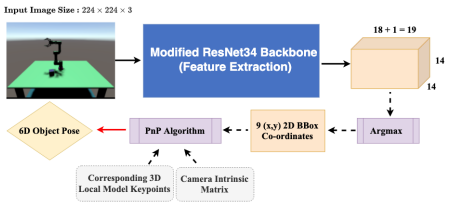

Fig. 4: Model Architecture of *Pose6DSSD*. It is a correspondence-based approach, which involves estimation of the 2D keypoints followed by extraction of pose information using the Perspective and Point (PnP)[6] algorithm.

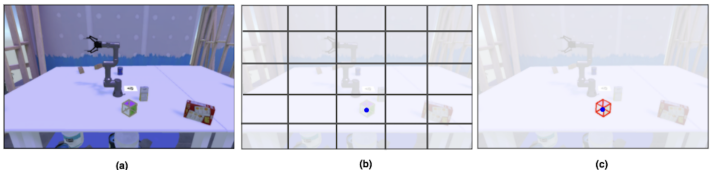

Fig. 5: (a) An example input RGB image. (b) The image is divided into $S \times S$ regions denoted by the square grid. (c) Each cell predicts 2D locations of the corners of the projected 3D bounding box in the image.

3.2.1 Interpreting Feature Extraction Output For a single image input, the output of the main feature extraction backbone is 3D tensor of dimensions $S \times S \times (2K+1)$. The input image is partitioned into a 2D regular grid (see Fig. 5) with $S \times S$ cells. In this work, $S = 14$ is considered.

For each grid, $2K+1$ values are predicted where K is the number of keypoints being considered. We consider a 3D bounding box based approach where $K = 9$ (8 corners + 1 centroid). The remaining one value predicts the confidence value of the grid cell, i.e. how confident the model is that in a given grid cell the object of interest is present. Note that, the model predicts the normalized offset values from the bottom-left grid point of each grid cell, similar to [11].

3.2.2 Modelling the Ground Truth Confidence As proposed in [11], the ground truth confidence values for training the model is modelled with an exponentially decreasing profile. The intuition behind this is the idea that the

confidence value is low when there is no object in grid cell, it is high when the object is present in the grid cell.

After feature extraction, the grid cell output with the maximum confidence value is selected as the final candidate. These 2D image coordinate predictions along with the corresponding 3D model points expressed in the local model frame and camera intrinsic matrix, form the input to the PnP block, whose output is the final predicted 6D pose for the object of interest present in the input image.

3.3 Model-3 : DOSSE-6D

We can observe that the previous model used a traditional correspondence based (using DL), so it is not an end-to-end approach, as we cannot directly utilize the output 6D pose information for training the model. We rely on indirect supervision (see Section 4) for training the model. The third model *DOSSE-6D*[4] is an improvement of the previous model in this aspect. Three versions of this model have been developed, each of them having small differences, compared to the other. More details are provided in Supplementary Material. The architecture of *DOSSE-6D_ v2* is illustrated in Fig. 6.

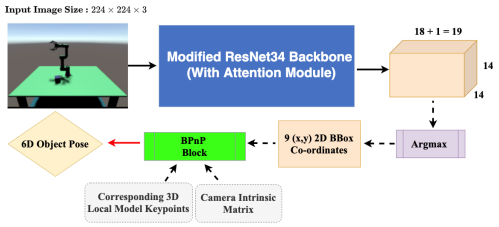

Fig. 6: High Level Model Architecture of *DOSSE-6D_ v2*. It is a correspondence-based approach, which involves estimation of the 2D keypoints followed by extraction of pose information using the BPnP module. Input is an RGB image of resolution 448×448 px.

DOSSE-6D is also a correspondence based approach similar to the Pose6DSSD, but with the following modifications :

- PnP Block is replaced by BPnP (Backpropagatable PnP) [2] module to make the model end-to-end trainable.
- The versions 2 and 3 of the *DOSSE-6D* utilize attention modules in their architectures for adaptive feature refinement of intermediate feature maps.

3.3.1 BPnP Module The BPnP block, as proposed in [2], is a module that backpropagates gradients through a PnP "layer" to guide parameter updates of a neural network. Based on the concept of implicit differentiation, it helps

[4] Stands for **D**eep **O**bject **S**ingle **S**hot **E**stimator of **6D** object pose

to combine DL network and geometric vision to form an end-to-end trainable pipeline. As suggested in [2], we define a stationary constraint function such that the output pose is local minimum for PnP solver. Then using the Implicit Function Theorem (IFT) and chain rule we can compute the necessary gradients for backpropagation. Using this (instead of Fully connected layers as in Model-1) we can optimise feature based loss and learn geometric constraints in an end-to-end manner. More mathematical details are provided in Supplementary Material.

3.3.2 Attention Module The main idea behind the use of attention module is to to force the model to focus on important features and suppress unnecessary ones, thereby improving its hidden representations. Incorporation of Attention module was done by the considering the best empirical practices in [15] and [14], found by extensive experimentation. Each attention module basically consists of two sub-modules namely :

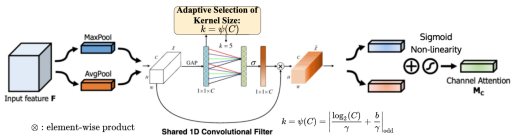

Fig. 7: Channel Attention Sub-Module Architecture. The intuition behind using both the type of features is that Max-pooled features encode the degree of the most salient part in the feature map, whereas Average-pooled features encode global statistics softly.

- *Channel Attention* – The intuition behind channel attention sub-module is to improve the feature maps by cross-channel interaction. One way that can be done is by selectively weighting each feature channel adaptively. We use Efficient Channel Attention (ECA) block, proposed in [14], which uses a 1D convolution, hence limiting the number of parameters. As shown in Fig. 7, in this work, both the Maxpool and Average pool features (pooling performed along the spatial dimensions, to get a 1D vector of length equal to number of channels) are passed through a shared 1D convolutional layer.
- *Spatial Attention* – This submodule computes a spatial attention map is obtained which can be used to improve features utilizing the inter-spatial relationship of features. As shown in Fig. 9, we are using both the Max-pool and Average pool features. Both these feature maps are stacked together and 2D convolution is performed followed by passing the output through a sigmoid non-linearity to restrict the range of values to $[0, 1]$.

For relative placement of submodules, as discussed in [15], a series configuration with channel attention sub-module preceeding the spatial attention sub-module is considered. Since, we are using a ResNet adapted backbone in all

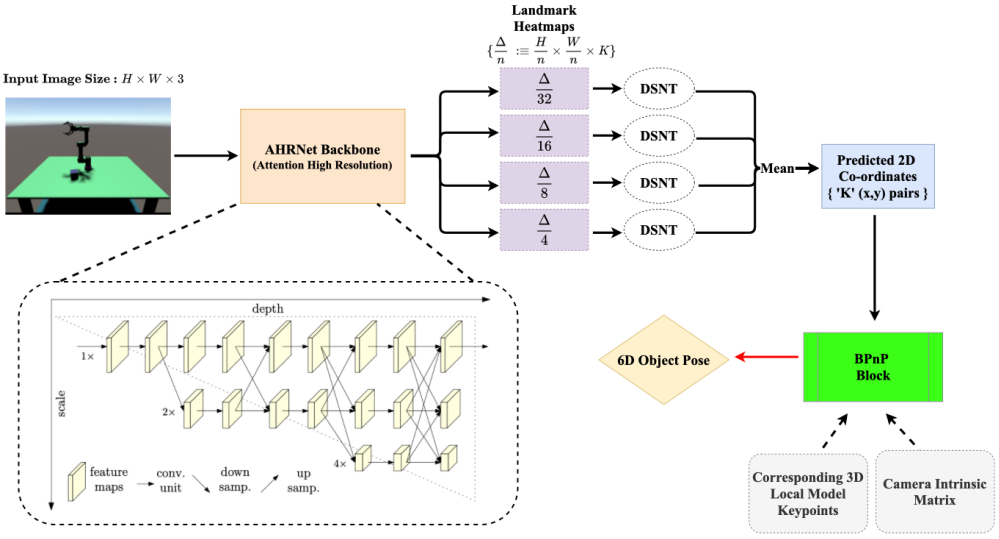

Fig. 8: High-Level Block Diagram of *AHR-DOSSE-6D*. It is a correspondence-based approach, which involves estimation of the 2D landmark heatmaps with keypoints, followed by extraction of pose information using the BPnP module.

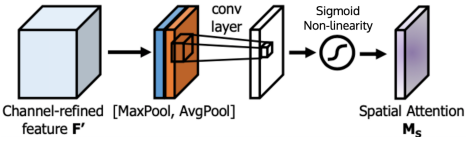

Fig. 9: Spatial Attention Sub-Module Architecture. This sub-module helps the model to basically decide "where" to focus in a feature map.

versions of *DOSSE-6D*, the complete attention module is placed at the end of each ResBlock (consists of two convolutional layers before the skip connnections).

3.4 Model-4 : AHR-DOSSE-6D

The last model designed is *AHR-DOSSE-6D*[1], whose high level architecture is illustrated in Fig. 8. It is a correspondence-based approach, which involves estimation of the 2D landmark heatmaps with keypoints, followed by extraction of pose information.

Some of the important elements of this model are as follows :

– Single Stage end-to-end trainable model (due to the use of BPnP module discussed in Section II.C) without any post-refinement stages.

[1] **A**ttention **H**igh **R**esolution **D**eep **O**bject **S**ingle **S**hot **E**stimator of **6D** object pose

- Use of attention module consisting of both spatial and channel attention sub-modules (See Section 2.3).
- Maintenance of high resoultion feature representations throughout the *AHR-Net* feature extraction backbone.
- Use of more geometrical details of the object under consideration by considering Farthest Point Sampling (FPS)[8] instead of 3D bounding box corners.
- Modelled confidence approach as described in Section II.B is replaced with heatmap estimation. The landmark heatmaps are used to obtain the 2D coordinates in normalized form, using the Differential Spatial to Numerical Transform (DSNT) block.
- Provision to use increased image resolution without increase in number of parameters.

1. *AHRNet Backbone:* The complete architecture of *AHRNet* backbone is provided in Supplementary Material. The main idea is the use of repeated multi-scale fusions to improve quality of hidden representations. Note that, as shown in Fig. 8, along depth axis feature map size remains same and along scale axis, typical feature map size reduction happens similar to a typical CNN.

2. *DSNT Block:* The Differentiable Spatial to Numerical Transform (DSNT) [7] block, in simple words, is used for converting the landmark heatmaps produced by the AHR-Net Backbone to normalized 2D coordinates of the corresponding keypoints. It is basically a spatial "soft"-argmax over each feature channel. Some of its desirable properties are that it is fully differentiable thereby helping in end-to-end design, adds no trainable parameters, exhibits good spatial generalization and it also performs well even with low heatmap resolutions.

4 Training Configuration and Model Evaluation

In this section, different attributes of the training configuration utilized such as experimental setup, loss functions and optimizer details are described.

4.1 Experimental Setup

Two simulation scenes were considered namely, *Simple Scene* and *Cluttered Scene* for collecting domain randomized, labelled data (see Fig. 10). Note that, *Cluttered Scene* is a more challenging version of the simple scene. For both the scenarios, data split used for all models is $\{$ *Training* | *Validation* | *Test*$\}$: $\{30000 \mid 3000 \mid 3000\}$ RGB images. For testing the performance of the model for estimating object pose, we consider the following both Same-Environment and Cross-Environment cases. Same-Environment cases test the model performance with respect to the generalizability of 6D pose predictions in a *known* environment, by training and testing the model in the same scene. Cross-Environment cases test the model's robustness and its ability to generalize over both the environment and the 6D pose predictions.

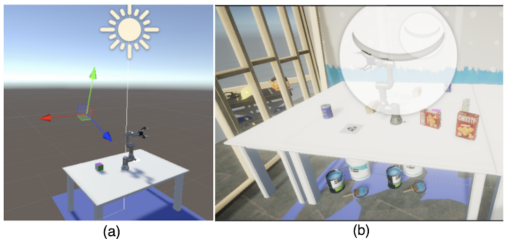

Fig. 10: Stills from Simulation scenes. (a) Simple Scene - Coloured face cube (obj. of interest) kept on the table where UR3 Robotic Arm is mounted. (b) Cluttered Scene - typical room environment with various objects placed at random positions (3D models of the distractor objects are taken from the YCB model set[1]

4.2 Loss Functions

In this subsection, more details about the loss functions utilized for training the corresponding pose estimation models discussed in Section III will be given. For *Model-1 : UnityVGG16*, the loss function considered is :

$$\mathcal{L}_1 = \mathcal{L}_{trans} + \mathcal{L}_{orient}$$

, where $\mathcal{L}_{trans}$ and $\mathcal{L}_{orient}$ are the Mean Squared Errors (MSE) between the predicted & ground truth translational vectors and the vectors representing the predicted & ground truth quaternions. The loss function utilized for *Model-2 : Pose6DSSD* is :

$$\mathcal{L}_2 = \lambda_{reproj}\mathcal{L}_{reproj} + \lambda_{conf}\mathcal{L}_{conf}$$

, where $\mathcal{L}_{reproj}$ is the MSE between the predicted and true projected 2D image point (normalized) coordinates and $\mathcal{L}_{conf}$ is the MSE between the predicted and ground truth confidence values. As proposed in [11], λ_{conf} is region-selective (more importance given to grid cells with object of interest) whereas $\lambda_{reproj} = 1$ everywhere. The mixture loss function considered for *Model-3 : DOSSE-6D* is given by:

$$\mathcal{L}_3 = \overbrace{\lambda_{reproj}\mathcal{L}_{reproj} + \lambda_{conf}\mathcal{L}_{conf}}^{\text{Indirect Supervision}} + \overbrace{\lambda_{add}\mathcal{L}_{add}}^{\text{Direct Supervision}}$$

where the definitions of L_{reproj} and $\mathcal{L}_{conf}$ are same as earlier. $\mathcal{L}_{add}$ is the average squared distance of 3D model points $\mathbf{x} \in \mathcal{M}$, between predicted and ground truth configurations of the object.

$$\mathcal{L}_{\text{add}} = \frac{1}{m} \sum_{\mathbf{x} \in \mathcal{M}} \|(\mathbf{R}\mathbf{x} + \mathbf{T}) - (\tilde{\mathbf{R}}\mathbf{x} + \tilde{\mathbf{T}})\|^2$$

where $\mathbf{R}, \tilde{\mathbf{R}}$ are Ground Truth and Predicted Rotation Matrices and $\mathbf{T}, \tilde{\mathbf{T}}$ Ground Truth and Predicted Translational Vectors respectively.

For *Model-4 : AHR-DOSSE-6D*, the loss function is given by :

$$\mathcal{L}_4 = \overbrace{\lambda_{reproj}\mathcal{L}_{reproj} + \lambda_{heat}\mathcal{L}_{heat}}^{\text{Indirect Supervision}} + \overbrace{\lambda_{add}\mathcal{L}_{add}}^{\text{Direct Supervision}}$$

where the definitions of L_{reproj} and L_{add} remain the same. $\mathcal{L}_{heat}$ (representing multi-scale supervision) is basically the MSE between predicted and ground truth heat maps :

$$\mathcal{L}_{\text{heat}} = \frac{1}{S}\sum_{s=1}^{S}\frac{1}{K}\sum_{k=1}^{K}\left\|\mathbf{H}_k^{s,pred} - \mathbf{H}_k^{s,\text{true}}\right\|_F^2$$

where

– The heatmap corresponding to the i^{th} keypoint and of scale (dimension) $\frac{\delta}{z}$ is :

$$\{\mathbf{H}_i^{j,pred}, \mathbf{H}_i^{j,true} \in \mathbb{R}^{\frac{\delta}{z}} \mid \frac{\delta}{z} :\equiv \left(\frac{H}{z} \times \frac{W}{z}\right), z = 4{\cdot}2^{j-1}\} \quad ; \quad i = 1, 2, \ldots, K \text{ and } j = 1, 2, \ldots, S$$

– $\|\cdot\|_F$ represents the Frobenius Norm of matrices. S represents the number of "scales" of heatmaps generated. In this work, $S = 4$ is considered.

4.3 Optimizer and Evaluation Details

For training all the pose estimation models, the *ADAM* optimizer is considered and checkpointing of the models is done at the epoch at which the model achieves best validation performance. The different evaluation metrics used for testing the object pose estimation performance of the different models are the popularly used Average Distance (ADD) of model points, Reprojection Error, Translational MSE and Quaternion Error (angular distance between ground truth and predicted quaternions).

5 Results

5.1 Unity Simulation Scenarios

Table 1 provides results for the test performance of the pose estimation models on the Unity Synthetic data. For all the results presented, the object of interest is a coloured face *cube* object of side Length (a) 10 cm.

S.No.	Approach \ Expt. Config.	Train-Clutter + Test-Clutter	Train-Clutter + Test-Simple	Train-Simple + Test-Simple	Train-Simple + Test-Clutter
1.	**UnityVGG16**	1.6801	16.5287	2.0248	53.7345
2.	**Pose6DSSD**	1.3976	9.0066	1.0054	39.0549
3.	**DOSSE-6D_v1**	1.2150	3.9213	0.9789	58.1505
4.	**DOSSE-6D_v2**	0.8836	10.5477	0.7604	41.8551
5.	**DOSSE-6D_v3**	0.9540	30.3129	1.0083	48.6070
6.	**AHR-DOSSE-6D**	0.4192	22.6130	0.4685	92.2395

Table 1: Table displaying the average ADD metric values (in cm). The model architectures and other details are described in previous sections. *v3* of *DOSSE-6D* uses low input image resolution whereas *v1* uses no attention module.

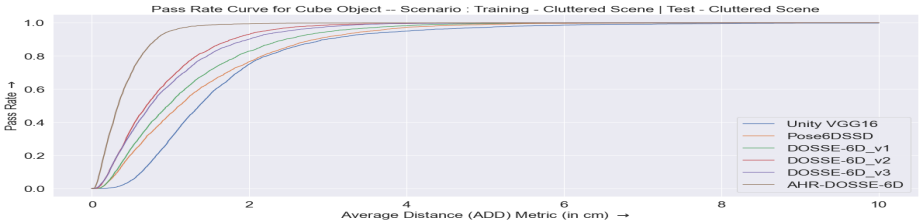

Fig. 11: ADD metric, plotted versus of pass rates (in %), for various models in Expt. Config. : Train-Clutter + Test-Clutter

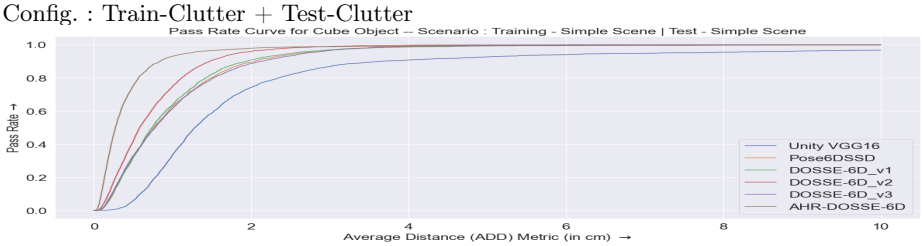

Fig. 12: ADD metric, plotted versus of pass rates (in %), for various models in Expt. Config. : Train-Simple + Test-Simple

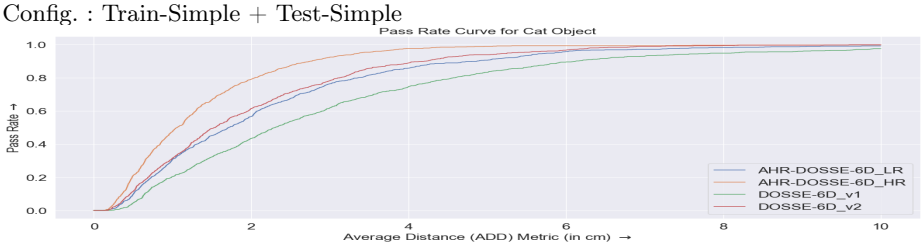

Fig. 13: ADD metric, plotted versus of pass rates (in %), for various models considered for the LINEMOD “cat" object.

5.2 LINEMOD Dataset

Table 2 provides the results for the ADD metric, in terms of pass rates (in %), for various models and objects in LINEMOD[4] dataset. For each model, 10% of

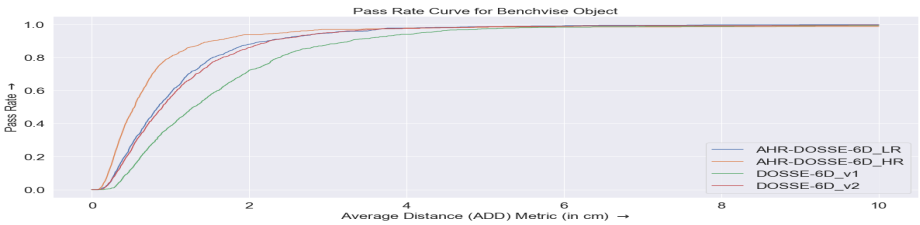

Fig. 14: ADD metric, plotted versus of pass rates (in %), for various models considered for the LINEMOD "benchvise" object.

the diameter (d) of object threshold for pass rate. Note that, input RGB image sizes of *DOSSE-6D_ v1*, *AHR-DOSSE-6D_ LR* is 224×224 and *DOSSE-6D_ v2*, *AHR-DOSSE-6D_ HR* is 448×448.

Some of the inferences extracted from the results of these test experiments are summarized below :

- We can observe that from results for the *Same Environment Cases*, the *AHR-DOSSE-6D* model performs the *best* as compared to the other models. The improvement in performance can be attributed to different elements in its design. Also, *AHR-DOSSE-6D_ HR* seems to give a relatively superior performance on all LINEMOD objects tested.
- In general, the performance of *DOSSE-6D* (all versions) models is higher than that of *Pose6DSSD* and *UnityVGG16* model due to the use of *Direct Supervision*.
- It can be seen that the all the models perform poorly as compared to the *Same Environment Cases* as expected, due to the fact that it is a much difficult problem. Here, essentially the "transfer" of pose prediction performance across environments is being tested.
- As expected the generalization capability of the models trained in *Cluttered Scene* is higher, i.e. these models are more robust to changes in environment, due to presence of background distractor objects.
- It is clear by observing the performance of the model pairs *DOSSE-6D_ v1* & *DOSSE-6D_ v2* and *AHR-DOSSE-6D_ LR* & *AHR-DOSSE-6D_ HR*, the models which take high input resolution image as input generally tend to perform much better than their counterparts.
- Transfer Learning is utilized in models *UnityVGG16*, *Pose6DSSD*, *DOSSE-6D_ v1*. Specifically, all these models consisted of a ResNet34 based feature extraction backbone, whose weights were initialized by pretraining on the ImageNet dataset for classification (a different task). Interestingly, it can be observed such pretraining is highly beneficial for improving generalization ability of the model across environments and leads to improvement in the *Cross Environment* performance of the models.

6 Conclusion

This work is aimed at developing robust, efficient and effective object 6D pose estimation techniques. A complete end to end pose estimation pipeline where a

S.No.	Approach \ Object	Cat (d = 15.50 cm)	Benchvise (d = 28.69 cm)	Lamp (d = 28.52 cm)	Can (d = 20.20 cm)	Iron (d = 30.32 cm)
1.	**SSD-6D** [5]	0.51	0.18	8.20	1.35	8.86
2.	**Tekin et al.** [11]	41.82	81.80	71.11	68.80	74.97
3.	**DOSSE-6D_v1**	33.45	86.77	74.94	60.19	60.22
4.	**DOSSE-6D_v2**	50.23	94.53	85.55	78.01	82.45
5.	**AHR-DOSSE-6D_LR**	45.89	94.30	94.36	84.14	88.10
6.	**AHR-DOSSE-6D_HR**	68.31	96.69	97.86	95.02	93.63

Table 2: Table displaying the ADD metric pass rates (in %). The rows highlighted in grey colour are some of the popular existing methods that provide results for*single stage, single object, single instance* object pose estimation using RGB images only.

UR3 robotic arm is deployed in a simulated pick and place task is demonstrated. The majority of this work has been focused at developing improved CNN based models for estimating pose from a single RGB image, utilizing neither depth information nor post hoc refinement stages. A series of such pose estimation models are designed, compared and analyzed. In order to test the efficacy of the approaches, they are trained and tested on synthetic data from simple and cluttered scenes, in a same-environment and cross-environment setting. The developed models were also tested on various objects from the LINEMOD benchmark dataset and the results indicated superior performance of the models. An interesting extension of this work is developing an improved pose estimation approach for Multi-object, Multi-instance prediction. One way of doing this is the inclusion of Part Affinity Fields (PAFs) prediction component into the AHR-DOSSE-6D model. Incorporation of additional collision avoidance modules to develop a more robust "safety-critical" approach to minimize the collisions, while executing the planned robotic arm trajectory is also worth exploring.

Supplementary information

Code and supplementary material is available upon request.

Bibliography

[1] Calli, B., Walsman, A., Singh, A., Srinivasa, S., Abbeel, P., Dollar, A.M.: Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set. IEEE Robotics Automation Magazine **22**(3), 36–52 (2015). https://doi.org/10.1109/MRA.2015.2448951

[2] Chen, B., Parra, A., Cao, J., Li, N., Chin, T.J.: End-to-end learnable geometric vision by backpropagating pnp optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8100–8109 (2020)

[3] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
[4] Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) Computer Vision – ACCV 2012. pp. 548–562. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
[5] Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In: Proceedings of the IEEE international conference on computer vision. pp. 1521–1529 (2017)
[6] Lepetit, V., Moreno-Noguer, F., Fua, P.: Epnp: An accurate o (n) solution to the pnp problem. International journal of computer vision **81**(2), 155 (2009)
[7] Nibali, A., He, Z., Morgan, S., Prendergast, L.: Numerical coordinate regression with convolutional neural networks. arXiv preprint arXiv:1801.07372 (2018)
[8] Peng, S., Zhou, X., Liu, Y., Lin, H., Huang, Q., Bao, H.: Pvnet: pixel-wise voting network for 6dof object pose estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020)
[9] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
[10] Tejani, A., Tang, D., Kouskouridas, R., Kim, T.K.: Latent-class hough forests for 3d object detection and pose estimation. In: European Conference on Computer Vision. pp. 462–477. Springer (2014)
[11] Tekin, B., Sinha, S.N., Fua, P.: Real-time seamless single shot 6d object pose prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 292–301 (2018)
[12] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS). pp. 23–30. IEEE (2017)
[13] Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., Birchfield, S.: Deep object pose estimation for semantic robotic grasping of household objects. arXiv preprint arXiv:1809.10790 (2018)
[14] Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., Hu, Q.: Eca-net: Efficient channel attention for deep convolutional neural networks. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 11531–11539 (2020)
[15] Woo, S., Park, J., Lee, J.Y., Kweon, I.S.: Cbam: Convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018)
[16] Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199 (2017)