3D Pose Estimation based on Deep Learning without Real World Data Training

Chen-Ruei Sin and Wei-Liang Lin
National Chung Hsing University, Taichung City, Taiwan, ROC
rayxin310@gmail.com and wllin@nchu.edu.tw

***Abstract*—This paper proposes a new methodology to carry out the 3D pose estimation of an object. We assume a single lens camera environment, and use a game engine to generate virtual world datasets in order to assist a deep learning model. Using only the virtual world datasets to conduct training has been tested with the real world data and the accuracy can be achieved up to 84.22% with 30M parameters.**

I. INTRODUCTION

The rapid development of deep learning in recent years has attracted more researchers to study machine vision, leading the increasing number of research papers on angle estimation. The accuracy is often affected by the size and reliability of datasets since the datasets are often labeled manually. For instance, Pascal3D+ uses the 3D CAD model to mark Ground Truth. Yet, relying on this manual labeling method is prone to inaccuracy when it comes to reality. The actual size of any car is different from that of any model. Hence, it takes a lot of manpower and it is time consuming to rectify deviations/differences caused by the manual labeling.

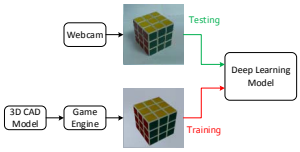


Figure 1. Flowchart

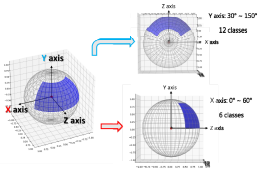


Figure 2. Angle Distribution

We propose a new approach of a game engine to automatically generate virtual world datasets, ensuring that the generated data has sufficient reliability. The virtual world datasets were later optimized with deep learning network training model which was directly tested in the real world datasets (Figure 1).

Our network function was limited to the spatial angle estimation of a single object, and the angle type was based on the left-hand rule. The angles of the two axes were measured, including the X axis and the Y axis (Figure 2).

As shown here, the estimated value was acquired at the relative angle between the observer and the target. Figure 2 shows that the target was located in the center of the sphere, and the observer moved on the sphere.

II. RELATED WORK

Sometimes, datasets were the images taken directly from those rendered by the game engine. This is the method proposed by Stephan R. et al.[1] who used this method to train image processing related tasks. However, they used the commercial games to extract G-Buffer and other related information, again relying on the manual labeling, despite the fact that the quality of its datasets is better than that of the existing semantic segmentation datasets. Our current study was inspired by their method.

There are two main approaches to estimate spatial angles. One is to estimate an angle as a regression problem[2][3], and the other is to divide an angle into multiple categories, as a classification problem[4]. In our research, we used the method of dividing an angle into multiple categories. If we treat the angle as a regression problem, the neural network will be difficult to converge and may require a larger network to extract features.

III. DATASETS

We used the Unity3D game engine to assist us in generating virtual world datasets. Our target is a magic cube because its cube has a fixed rotation axis on any axis, and it rotates clockwise or counterclockwise every 90 degrees, with the same shape. We believe it raises certain difficulty while estimating the angles.

We also added lighting effect to the scene to simulate the light and shadow effect in the real world. It only took about 10 minutes to generate a more than 20,000 pieces of dataset with detailed angle information. The dataset was later extracted the corresponding categorized information.

In addition to the virtual world datasets, we collected real world datasets to test the accuracy of our deep learning model. In order to ensure that the accuracy of angle captured in the real world, we specially designed a shooting environment (Figure 3).

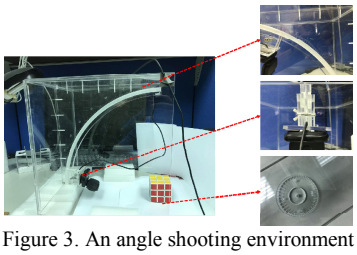


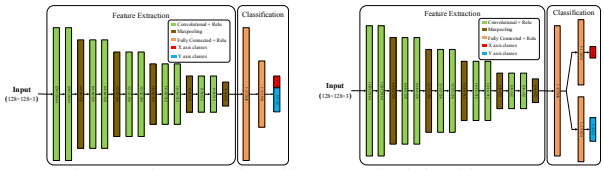
Figure 3. An angle shooting environment

Authorized licensed use limited to: National Chung Hsing Univ.. Downloaded on July 22,2021 at 06:21:51 UTC from IEEE Xplore. Restrictions apply.
978-1-7281-7399-3/20/$31.00 ©2020 IEEE

There were 20,000 training samples generated by the virtual world, and 2,000 verification samples. The testing samples include 1,000 generated in the virtual world, and 864 gathered from the real world.

IV. Network Architecture

Our network structure is mainly based on VGG16[5], used as the prototype of the feature extraction layers. For the latter classification, two different structures were used for experiments (Figs. 4 & 5). We were inspired by some of the ideas from [4]. The overall structure has been simplified to reduce the weights.



Figures 4./5. Network architecture: Single/Multi-Output

Fig. 4 depicts the network architecture which estimated the rotation angles related to the X and Y axes. VGG16 was used as the prototype for the feature extraction layers with the reduced number of parameters. In terms of the classification layers, the outputs were combined so each output had two maximum values (X and Y). This technique could be used to compress the parameters to smaller values. This network structure is referred as Single-Output.

The feature extraction layers in Fig. 5 is very similar to that of Fig. 5. The main difference lies in the classification layers. We adopted the concept proposed by[4]. but separated each axis as an independent variable for classification. This network structure is referred as Multi-Output.

In terms of the loss function, the Single-Output network architecture had multiple maximum values so binary cross-entropy loss was used. The Multi-Output network architecture had only one maximum value for individual outputs while classifying the X-axis and Y-axis angles respectively, so the categorical cross-entropy loss function was used for evaluation.

V. Experiments

The experiments have the input image size of 128*128*3. A total of 500 epochs was executed.

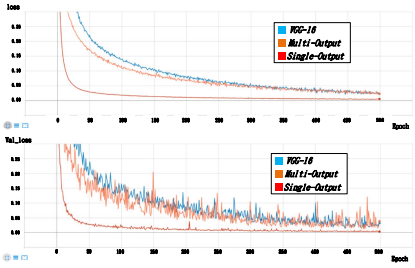


Figure 6.Training & Validation loss trends

The three different deep learning models were performed, including VGG-16 and the two VGG-16 improved deep learning models (Multi-Output and Single-Output). Fig. 6 indicates that the speed for Single-Output convergence was the fastest, and the speed for the Multi-Output and VGG-16 convergence was slower.

Upon the completion of the training, the virtual world datasets and the real world datasets were tested separately. The test results are shown in the following Table 1 and Table 2 respectively.

Table 1.Virtual world data test

	Single-Output	*Multi-Output*	*VGG-16*
X-Accuracy	99.80%	**99.90%**	99.80%
Y-Accuracy	**99.40%**	99.10%	99.00%
Accuracy	**99.53%**	99.37%	99.26%

Table 2.Real world data test

	Single-Output	*Multi-Output*	*VGG-16*
X-Accuracy	74.31%	**89.00%**	85.08%
Y-Accuracy	62.34%	**81.83%**	81.34%
Accuracy	66.33%	**84.22%**	82.58%

In Table 1 and Table 2, the confusion matrix was first used to calculate the accuracy. Finally, a weight function was added to the predicted accuracy of the X-axis and the Y-axis.(Eq.1) Y-axis has a higher weight due to a higher number of catagories.

$$Accuracy = \frac{X_{Accuracy} + 2(Y_{Accuracy})}{3} \quad (1)$$

It is found that the accuracy of each architecture using the virtual world datasets was achieved up to 99%, and the accuracy of using real world datasets with Multi-Output model was achieved up to 84.22%. The numbers of parameters in our architectures are 30M and 28M for Multi-output and Single-output, respectively. The VGG-16 architecture has 65M parameters.

VI. Conclusions

We have introduced how to use the game engine to perform the automatic virtual world dataset generation to assist the training in deep learning models in terms of 3D pose estimation. Also, we have proposed different methods to improve the structure of deep learning models to make it more light-weighted while achieving a better accuracy.

VII. References

[1] S. R. Richter, V. Vineet, S. Roth, and V. Koltun., "Playing for data: Ground truth from computer games.," European Conference on Computer Vision, pages 102–118. Springer, 2016.

[2] S. Mahendran, H. Ali, and R. Vidal, " 3D Pose Regression using Convolutional Neural Networks," CVPRW, 2017.

[3] A. Kendall, M. Grimes, and R. Cipolla, " PoseNet: A Convolutional Network for Real-Time 6-DOF camera relocalization," ICCV, 2015.

[4] J. Sochor, J. Špaňhel and A. Herout., "BoxCars: Improving fine-grained recognition of vehicles using 3-D bounding boxes in traffic surveillance,"

[5] K. Zisserman and A. Simonyan, "Very deep convolutional networks for large-scale image recognition," ICLR, 2015.

Authorized licensed use limited to: National Chung Hsing Univ.. Downloaded on July 22,2021 at 00:29:51 UTC from IEEE Xplore. Restrictions apply.