

# Visualization of Furniture using CNN for Augmented Reality

<sup>[1]</sup>Nidhi Singh, <sup>[2]</sup> Mahima Khawale, <sup>[3]</sup> Niranjana Dubule, <sup>[4]</sup> Dr. Sandeep M. Chaware

<sup>[1][2][3]</sup> B.E. Dept. of Computer Engineering, Department of Computer Engineering, Marathwada Mitra Mandal's College of Engineering, Pune, India

<sup>[4]</sup> Professor, Dept. of Computer Engineering, Department of Computer Engineering, Marathwada Mitra Mandal's College of Engineering, Pune, India

---

**Abstract---** Augmented Reality allows integration of 3D virtual objects into the real environment in real time thus allowing users to relate with their physical environment and making the experience more interesting. The manual conversion of the 2D images to 3D models is a tedious and time-consuming process. Hence, in this project, we automate the process of reconstruction of 3D models from 2D images using Convolutional Neural Networks (CNN). We develop a Depth-based image in the initial stage combined with CNN followed by the structural recovery and creating prediction to achieve the desired model using another CNN model. The reconstruction starts with less resolution which grows iteratively. Lastly, they are optimized using Stochastic Gradient Descent which makes them fit for use in Augmented Reality. This process of automated 2D-3D conversion is integrated with marker-less implementation of Augmented Reality through an Android Application to ultimately visualize furniture in the real world.

**Keywords—** Augmented Reality, Deep Convolutional Neural Networks, 2D-to-3D Conversion, Android Application

---

## I. INTRODUCTION

The use of Augmented Reality and Virtual Reality is increasing daily. The recent past has seen phenomenal growth of Augmented Reality and has attracted academia as well as industrialists. It seamlessly blends virtual reality with the real world and is being used in various quarters. In recent times we saw the emergence of Augmented Reality Mobile Applications like Google Glass, Microsoft HoloLens, etc. & the evolution of powerful Augmented Reality development kits like ARCore, ARKit, Vuforia, etc. combined with improved performance[1]. Augmented Reality has offered prominent advantages in fields of technology, entertainment, retail, etc.[2].

One of the initial problems faced was depth estimation for a single 2D image. The quality of depth estimates is crucial. Multi-view stereo images are used to recover the depth by calculating the approximate of the object from different angles[3]. Although we intend to work on similar lines, we use all the images available in the 2D image repository for the object and select the suitable ones for depth recovery.

The step-by-step manual process of generating 3D models for Augmented Reality, in Unity 3D is the most popular one[4]. The models created are exported to an application that is used for selling furniture. But the drawback is that all the manual work required for model generation hence limits the scope. Further, researchers have used Machine Learning to generate 3D models for the application for

feature extraction, feature learning, and texture estimation steps[5]. The advancement hence made still needs human attention at each step and the process also consumes a lot of time. 3D reconstruction from 2D images by using Convolutional Neural Networks (CNN) produces small 3D models with exceptionally low accuracy rate[6]. The models generated need optimization so they can be used efficiently[7].

Techniques to optimize 3D models for small-scale deployment specifies this, but their initial process time is more. They required a lot of human effort to accomplish the tasks, resulting in low scalability and accuracy for deployment. To eliminate all such limitations, we implement a system to develop a 3D model from a single 2D image of an object using a unified Deep Convolutional Neural Network (CNN) framework and achieve better accuracy. The generated models are optimized for use in Augmented Reality.

## II. IMPLEMENTATION

### A. Model Generation

#### a. Preprocessing

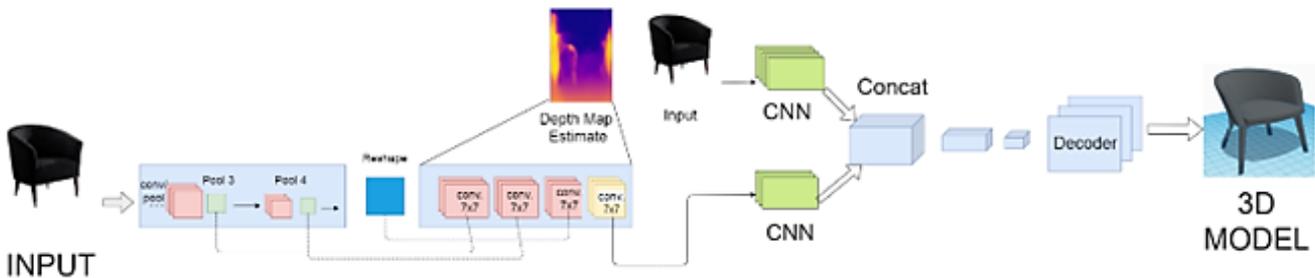
The process begins by creating an image-based furniture dataset by collecting images from various online resources. There are a vast number of categories of furniture that can be considered in the application. The scope of the dataset was set to certain parameters like image size, its quality, and the view of the furniture piece being displayed in the

image. Views like front or isometric have been incorporated and any type of furniture views that would hamper the depth mapping process in the further steps have been eliminated. The images have been broadly categorized into five types - Bedroom, Chairs, Sofas, Tables, Accessories and have been stored on a local database.

Before we start with the actual implementation, we need to preprocess the images to bring uniformity to the dataset.

For this purpose, manual background removal of all the images to remove all the unwanted features, converting the images to Grayscale, and resizing them has been done.

Then we convert them into OpenCV arrays that store the Grayscale intensity values from 0 to 255 (0 - white and 255 - black) for each pixel in an image. NumPy and OpenCV2 packages are used for these steps. After this preprocessing, we store these arrays as a training dataset.



**Fig. 1: Implemented Convolutional Neural Network System.**

Pseudo Code for preprocessing:

```

Store the dataset in a variable Furni_Dir
Store the categories in a variable CATEGORIES
Create empty training variable training_data
Img_height = 360
Img_width = 420
For category in CATEGORIES:
Convert each image into grayscale
Convert each image to Img_width x Img_height
dimensions
Append the converted image in training_data
End
    
```

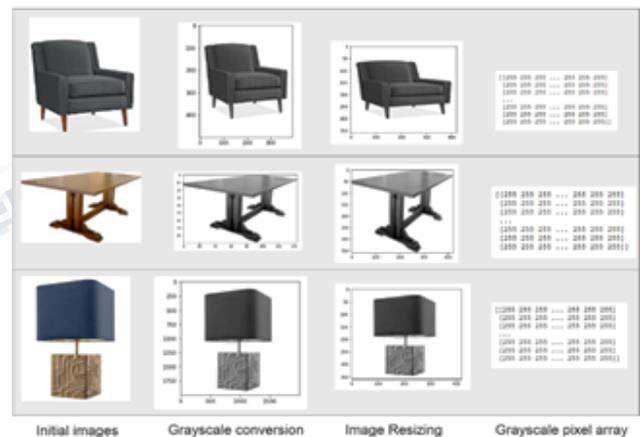
**b. Depth Estimation Network**

Depth estimation is perhaps the most straightforward solution for recovering 3D information from single images. Deep learning is to be highly effective for depth estimation. Compared to depth estimation, reconstructing a full 3D model is much more challenging due to texture, illuminance and the pattern of the object.

An auto-encoder is composed of two sub-networks: a depth estimation network for discerning the object structures from the input 2D image and a model generator network for a hierarchy of 3D boxes along with their mutual relations. Fig. 1 denotes the flow of our neural network system.

The depth estimation network is trained to estimate the contour (depth map) of the object of interest. This is motivated by the observation that object depth maps

provide strong cues for understanding shape structures in 2D images.



**Fig. 2: Flow of preprocessing from initial images to OpenCV array**

The first scale captures the information of the whole image while the second produces a detailed depth map. We initialize the convolutional layers of the first scale network, followed by two fully connected layers. The feature maps and outputs of the first scale network are fed into various layers of the second scale one for refined structure discerning. The second scale network, as a refinement block, starts from one 12x12 convolution and one pooling over the original input image, followed by

successive 7x7 convolutions without pooling.

### **c. Model Generator Network**

The model generator network integrates the features extracted from the depth estimation network and the input image into the bottleneck and decodes it into a hierarchy of part boxes. We combine features from two channels. One channel takes as input the depth map of the depth estimation network, accompanied by two convolutions and poolings. Another channel is the CNN feature of the initial image.

The output feature maps from the two channels are then combined with size 9x9 and further encoded after two fully connected layers, capturing the object structure information from the input image. Since it is hard for the depth estimation network to produce a perfect depth map, the CNN feature of the initial image provides extra information by collecting more object information.

The 3D models thus generated are stored on an online database i.e., Google Firebase. An online database would enable the Android application being developed to have easier and quicker access to these models whenever needed while being adequately storage efficient for the app user.

### **B. Back-end Processing for Augmented Reality**

When the user initiates the Augmented Reality feature in the mobile application, a series of operations take place. The first operation, in this case, is capturing and analysing the environment where the user wants to view the furniture model, using a rear camera from the smartphone[8]. The use of Unity3D for the same in combination with ARCore helps us achieve this pretty easily[9][10].

The environment analysis is done by the Unity framework automatically by using the PlaneDetectionManager utility. Based on the environment analysis and using the smartphone's accelerometer and gyroscope, the system will suggest if there is sufficient space for the projection or the user must change the location.

During the camera stream capturing, we detect a plane for object placement. As the user moves his camera in his surroundings, the accelerometer, and gyroscope detect two points, hereafter known as hit locations on the plane, and calculate the distance between them depending on the speed of camera movement. The distances that are measured are temporarily stored as dimensions of the area under observation.

Once the dimensions are calculated, the next step is to visualize the furniture pieces. For this purpose, we retrieve the stored 3D models from Firebase. Then, the models are

converted into object Prefab components that, once created, can be readily used for further calls of a particular model. Prefab components are configured GameObjects that enable reusability. These Prefabs are then placed in the game view of the Unity framework. Once this has been completed, we integrate the dimension measurements, the current position of the camera stream, and the movements and the Prefab to accurately place the furniture piece.

### **C. User Interface**

The entire system is blended in an Android application developed using Flutter and connected to Firebase for retrieval of each item as required. The application will have a basic user interface for browsing items, payment gateways, and shipment services. It controls the main functionality of the application while providing the Augmented Reality facility to users on the click of a button. Users can Sign Up/Login to the application and browse items. Each item has a detailed description of the product and the "AR Try Out" feature which will show the data augmented on the run, which will include basic transformation like rotation, flip, and translation.

## **III. EXPERIMENTS**

The implementation of the system and the comparative analysis were done on a single NVIDIA GeForce 940MX. It is performed on a Windows machine with Intel Core i5 7th Gen 2.71 GHz in CPU mode.

We are using Stochastic Gradient Descent (SGD) for optimization. The learning rate is set to  $10^{-4}$  for the first 10 epochs keeping the batch size of 32 and gradually decreasing the learning rate to  $10^{-6}$  until the networks converge and the batch size is set to 16. The Weight Decay and momentum are kept at 0.01 and 0.9 respectively and are constant throughout the process. Start the dropout with a minimum of 0.1% and the maximum dropout is set to '1.0%'.

Background removal and resizing the images to bring uniformity within the dataset resulted in efficient training. The blend of Augmented Reality with Deep Learning through an application gives us multiple metrics to check the overall effectiveness of the system. We have evaluated our approach based on the following metrics:

1. **Quality:** The quality of the 3D models holds utmost importance. In the table below we have compared our system with few state-of-the-art systems which are based on conventional CNN and a few Manual 3D model generation techniques.
2. **Time:** Time required to generate these models through various methods. As low as possible.

3. Efficiency: The number of computational resources used by the algorithm to generate close to accurate results, considering that there are minor errors in the process.

We use SSIM i.e., the Structural Similarity Index to calculate the degradation of the quality of images. It is the measure between two windows  $x$  and  $y$  of common size  $n \times n$ . The formula for SSIM is:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (1)$$

Where,  $\mu_x, \mu_y = average(x, y)$

$\sigma =$  variances respectively

$c_1, c_2$  are variables to stabilize the division with weaker denominator.

Table 2 shows the SSIM values along with the figures that have been achieved using these SSIM values.

We have also used MSE to calculate the loss caused due to preprocessing the images. MSE, i.e., the Mean Squared Index

**Table 1: MSE values comparison table**

Parameters	MSE	Loss
Our System	0.189	

Compared System	0.165 – 0.317	
-----------------	---------------	---

**Table 2: Experimented SSIM values.**

Original RGB Image	
SSIM = 0.798	



**Fig. 3: Output of Augmented Reality Application. (a) Scene from a different camera of how the augmentation appears. (b) Kabino Wooden Dresser (W x H x D = 74.2cm x 127.0cm x 41.0cm). (c) Striped bench for patio (Dimensions: W x H x D = 178cm x 42cm x 43cm).**

is the average of the squared error for image enhancement. Its formula is denoted by:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2 \quad (2)$$

#### IV. RESULTS

To test the Unity3D platform for augmentation we gave it a few 3D models which we collected from an open-source platform in .obj format. Fig. 3 displays the models in Augmented Reality.

Table 1 shows our MSE values compared with values obtained using another method. They used their system to compare three different datasets to generate 3D models from single-view 2D RCG images using Generative Adversarial Network (GAN) with one auto-encoder. For phase one of this process includes Depth-Image generation and feature extraction[12].

### Conclusion

We have proposed a system that uses Depth-Image unified with Convolutional Neural Network framework to recover 3D shapes from single-view 2D RGB images. When compared to state-of-the-art processes our system gives better results qualitatively and quantitatively. All this wrapped up with Augmented Reality is then accessible to users through an Android Application. This will assist the customers to view the furniture piece virtually in physical space in real-time without buying the object. The main advantage is that it would save a lot of time and save money for the customers.

### V. FUTURE WORK

The database required to store all the 3D models is huge. So, having cloud services that are cost-effective and can store huge data will help the application to run smoothly. On a commercial level, we can have integration with not just reputed brands but also the local brands, which will increase product sales, promote small-scale business, and reduce return rates. Currently, the scope is limited to furniture, Augmented Reality technology can also be used to visualize real estate properties in real time, which will be revolutionary in the real estate industry

### REFERENCES

- [1] J. He et al., "The research and application of the augmented reality technology," 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, 2017, pp. 496-501, doi: 10.1109/ITNEC.2017.8284781.
- [2] Cao, Sheng & Wang, Qian. (2017) "Application and Prospect of AR Technology in E-commerce" 10.2991/emcs-17.2017.262.
- [3] Viyanon, Waraporn, Thanadon Songsuittipong, Phattarika Piyapaisarn and Suwanun Sudchid. "AR Furniture: Integrating Augmented Reality Technology to Enhance Interior Design using Marker and Markerless tracking." IIP'17 (2017).
- [4] Ms. Kirti Motwani, Shweta Sharma, Dhanashree Pawar, "Furniture Arrangement Using Augmented Reality" in International Research Journal of Engineering and Technology (IRJET), Volume: 04 Issue: 04, Apr -2017.
- [5] M. Khedwala, F. Momin, U. Pachhapure and S. Shaikh, "Analysis of Auto Generation of 3D Model Using Multiple 2D Graphics to Manifest Through Augmented Reality," 2018 International Conference on Smart City and Emerging Technology (ICSCET), Mumbai, 2018, pp. 1-5, doi: 10.1109/ICSCET.2018.8537310.
- [6] K. Sakai and Y. Yasumura, "Three-dimensional shape reconstruction from a single image based on feature learning," 2018 International Workshop on Advanced Image Technology (IWAIT), Chiang Mai, 2018, pp. 1-4, doi: 10.1109/IWAIT.2018.8369636.
- [7] L. Li, X. Qiao, Q. Lu, P. Ren and R. Lin, "Rendering Optimization for Mobile Web 3D Based on Animation Data Separation and On-Demand Loading," in IEEE Access, vol. 8, pp. 88474-88486, 2020, doi: 10.1109/ACCESS.2020.2993613.
- [8] Boonbrahm, Salin & Boonbrahm, Poonpong & Kaewrat, Charlee. (2020) "The Use of Marker-Based Augmented Reality in Space Measurement" Procedia Manufacturing. 42. 337-343. 10.1016/j.promfg.2020.02.081.
- [9] Amin, Dhiraj & Govilkar, Sharvari. (2015) "Comparative Study of Augmented Reality Sdk's" International Journal on Computational Science & Applications. 5. 11-26. 10.5121/ijcsa.2015.5102.
- [10] Kasetty sudarshan, Sneha. (2017). AUGMENTED REALITY IN MOBILE DEVICES.
- [11] Galabov, Miroslav. (2015), "A Real Time 2D to 3D Image Conversion Techniques" International Journal of Engineering Science and Innovative Technology. 4. 297-304.
- [12] Stigeborn, P. (2018). *Generating 3D-objects using neural networks* [Published bachelor's thesis]. KTH Royal Institute Of Technology.