AR/VR Based Furniture Marketplace

BTP Report

by

Sayam Kumar: S20180010158
Nitin Kumar Chauhan: S20180010119
Dasari Jayasree: S20180010047

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRICITY

9 Dec 2021

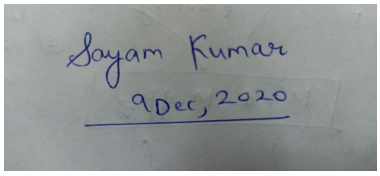
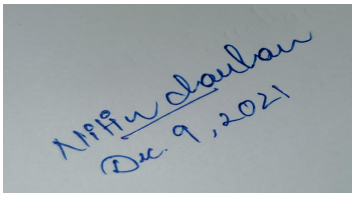
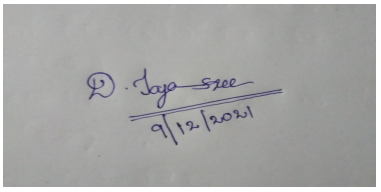
2nd Semester Report

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRICITY

CANDIDATE'S DECLARATION

We hereby certify that the work which is being presented in the BTP entitled "**AR/VR based Furniture Marketplace**" in the complete fulfillment of the requirements for the award of the degree of B. Tech and submitted in the Indian Institute of Information Technology SriCity, is an authentic record of my own work carried out during the time period from January 2021 to Dec 2021 under the supervision of Prof. Subu Kandaswamy, Indian Institute of Information Technology SriCity, India.
The matter presented in this report has not been submitted by me for the award of any other degree of this or any other institute.

Signature of the student with date

		
Sayam Kumar	Nitin Kumar Chauhan	Dasari Jayasree

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

14/12/2021

Signature of BTP Supervisor with date
(Prof. Subu Kandaswamy)

Abstract

Our BTP project idea is to use Augmented Reality (AR) / Virtual Reality (VR) to simplify the work of furniture showrooms and Interior designers. The prescribed project has two components. First, scan the furniture and generate a 3D model. Second, project a 3D model in Virtual space using Augmented Reality. The desirable outcome from our BTP project is a mobile app that provides both these components. Then this app can be used by furniture showrooms owners to increase customer engagement. Customers can get a feel of how a particular furniture will look like in their home before considering buying it. There is an acute shortage of mobile apps that work in the furniture marketplace. Our project is built to fill this market gap.

Contents

INTRODUCTION

Our project idea is to generate 3d models from input furniture images. Then to project these models in Virtual space using Augmented Reality. There is an acute shortage of mobile apps that work in the furniture marketplace. From Google play store, apps like DecorMate and HomeStyler have pre-built 3d models that are used for shopping purposes. These apps are constrained by limited options and have high pricing options for premium features. Furthermore, most of the tech giant companies have stopped investing heavily into using Augmented Reality and Virtual Reality based development. The prime investment area for development these days is in Artificial Intelligence (AI) and Machine Learning (ML). So, there is a void created in AR / VR apps during the recent years.

Our project is a good opportunity to build a first of its kind app. Let's debrief about both the components of our project - Generating a good visually appealing 3D model given an image is at the core of visual understanding. This itself is a two fold problem. We need to recover both the shape and color/texture from 2D images for building 3D models. Our proposed network "Parallel Voxel Network" (PVN) aims to recover both shape and color separately. Each Voxel Network uses Encoder Decoder networks that are built on convolution layers. We use sparse loss functions to generate visually appealing results. Then we convert 3d models to glb format and get it rendered in Virtual Space.

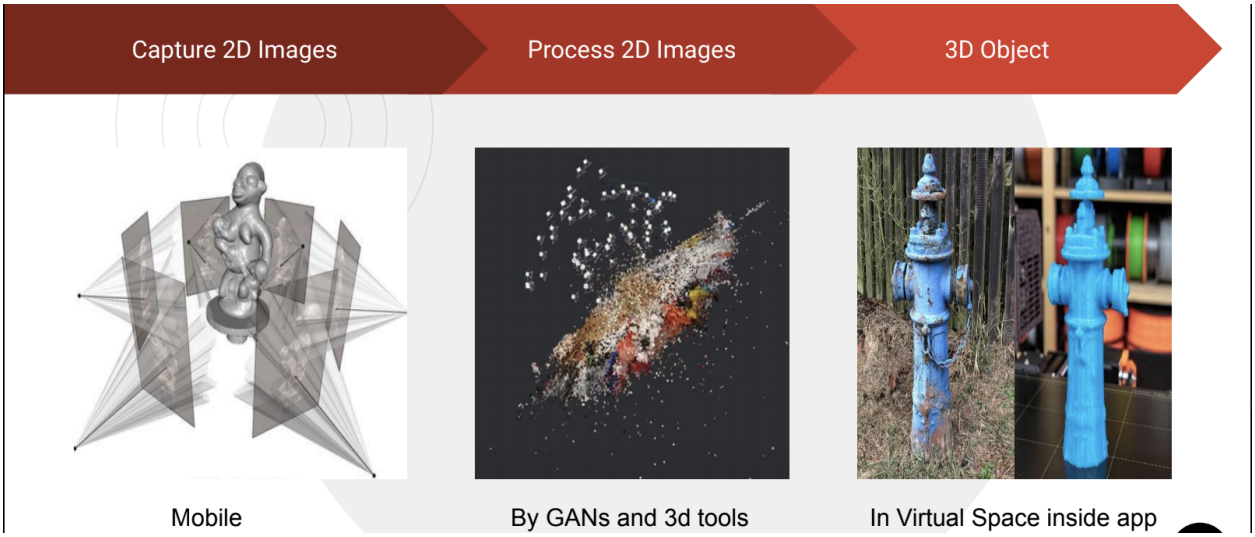


Figure 1 - Workflow of our mobile app

LITERATURE SURVEY

Existing works on modeling 2D images into 3D models mainly focus on shape recovery. But for the furniture marketplace, recovering color and surface textures are also equally important. Majorly literature revolves around two major ways for generating 3D models -

Geometric-based Reconstruction

The idea here is to make 3D objects from 2D images by satisfying geometry constraints and adding prior knowledge. Eg - If I am modeling a sphere, I make sure the 3D model has equal radius from center to surface (Moll et al, 2015). The other line of research is on scene-dependent designing of 3D models (SMPL et al, 2015). Then there are works on multiple view reconstruction (Zhou et al, 2016). Our work focuses on single image reconstruction.

Learning-based Reconstruction

These methods use data driven approaches for 2D-3D mapping. Initial works are described in 3D Recurrent Neural Network (3D-R2N2) (Choy et al, 2016) paper which uses multiple views of images and Long Short Term Memory (LSTM). (Zhu et al, 2017) generated 3d models from 2.5d inputs and made use of 3D U-Net networks. Further, 2D 3D enforced differential constraints were added to build visually appealing results. These data driven approaches make use of large scale datasets ShapeNet for 3D shape estimation.

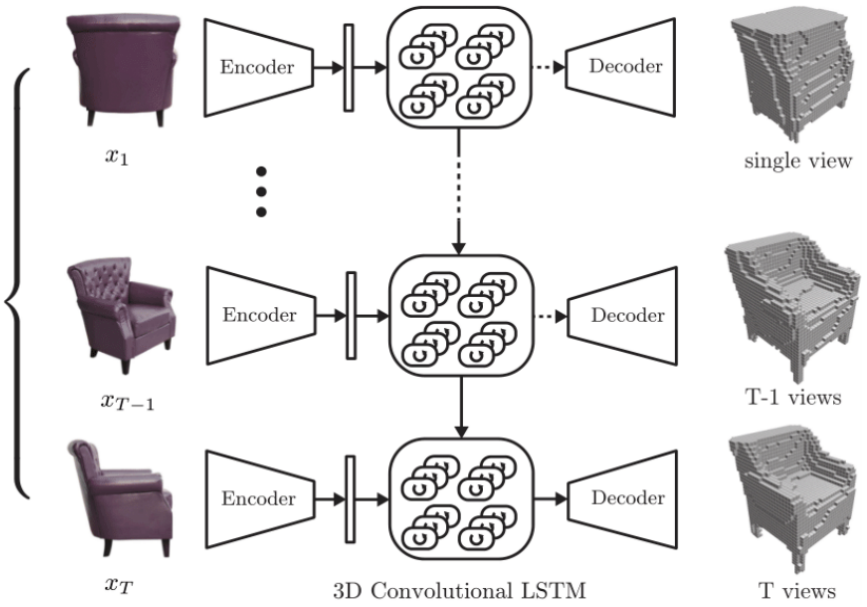


Figure 2 - 3D R2N2 approach

METHODOLOGY

This section defines all the methods we used to build the end-to-end mobile app.

Let's start by defining terminology to build the model:

Pixel

Smallest unit for a 2D image.
All images are formed by stacking and arranging pixels.

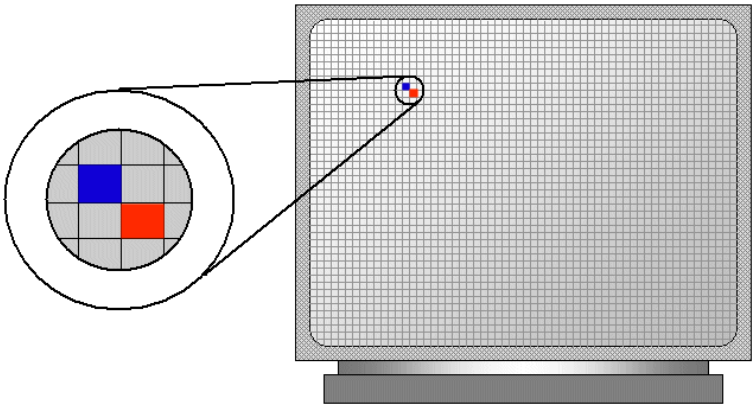


Figure 3 - Pixels in an image

Voxel

Smallest cubical unit for a 3d structure. All 3D structures are formed by stacking and arranging pixels.

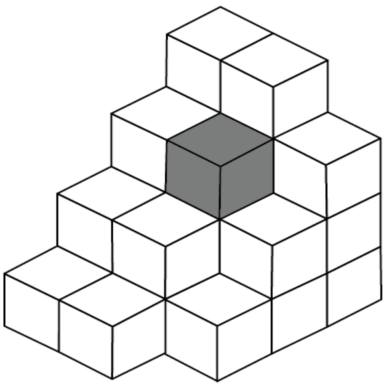


Figure 4 - Voxels in a 3D model

Problem Definition

Given an input Image (I), generate a 3D model (M) for it. This can be seen as a Supervised Learning problem. Let's say, we make use of the hypothesis function (h). M = h(I)

Further breaking it down, We have ground truth Image I and it's true model M from the dataset. Our network learns a transition from pixel representation of 2D image to voxel representation of 3D model. We compare the generated 3D model with the true 3D model and use sparse loss functions to train for visually appealing results.

Dataset

We are using the Shapenet dataset. Original shapenet is built by generating 3D models using Computer Aided Design (CAD). There 270 categories of objects and 51k unique 3D models. Our data pipeline has 2 categories - chair and table. We have 2.2k chair models, 2.5k table models, 1.4k cabinets, 2k sofa models. We generated 12 2D views for the 3D model as images captured by 30^{o} rotation around the vertical axis. We store images in png format and 3D models in obj format. The background from images is also removed if any.

Learning Representation

We need to learn two things from an image - 3D model shape and 3D model texture.

Joint Learning - The idea is to learn both tasks simultaneously. Each occupied voxel is learnt by 0 or 1. 0 means unoccupied voxel and 1 means occupied voxel. For color, the network learns three numbers for RGB for occupied voxels and a vector [-1, -1, -1] for unoccupied voxels. Learn 4 numbers in total. It turned out to be a bad approach because for sparse models (like chair) and imbalanced voxels, the network pushes more towards learning unoccupied voxels better.

Parallel Learning - Learn both tasks separately. This avoided the training difficulty due to shape color imbalance

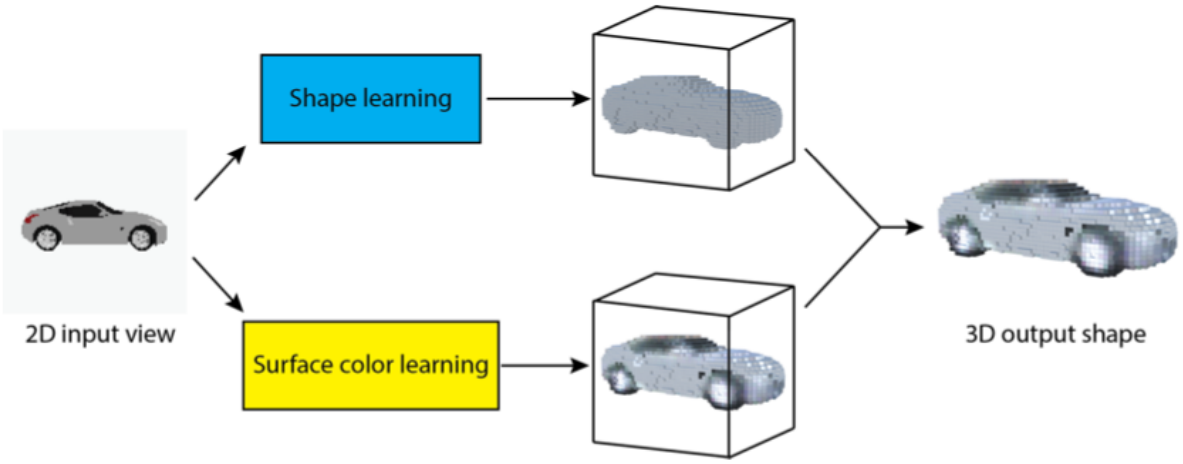


Figure 5 - Learning representation of our Parallel Voxel Network

Shape Network

Given an input image (I), the encoder network (e) first learns the latent representation and then is fed into a decoder network (d). The decoder generates a 1 channel shape volume V' = d(e(I))

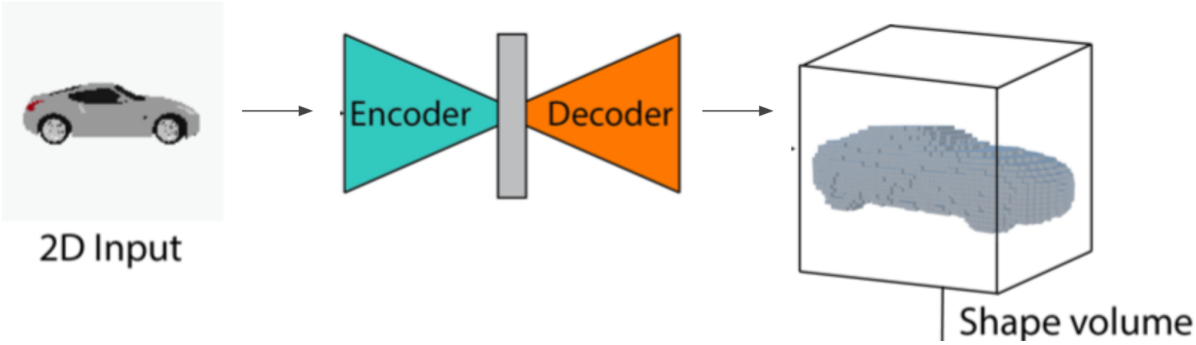


Figure 6 - Shape Network Pipeline

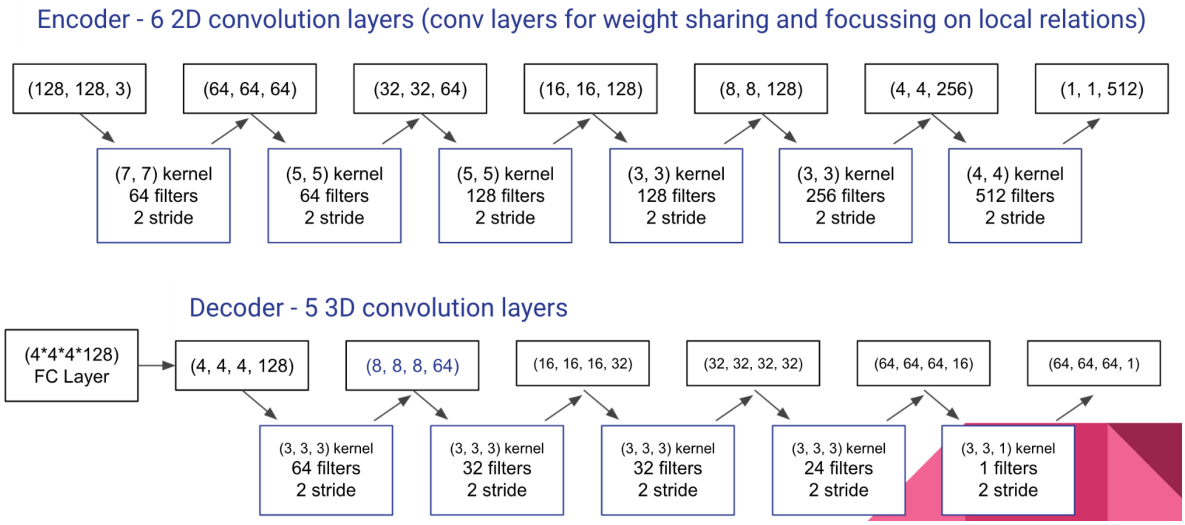


Figure 7 - Shape Network semantics

Loss functions

L2 loss = $\sum_i (V_i - \hat{V}_i)^2$, where i is voxel index and V_i is state of voxel occupancy {0, 1}

Moved to Cross Entropy Loss = $-\sum_i [V_i \log(\hat{V}_i) + (1 - V_i) \log(1 - \hat{V}_i)]$

To handle sparse relations better, we modified it to use two combined losses-

- False Positive Cross entropy on unoccupied voxels
- False Negative Cross entropy on occupied voxels

$$FPCE = -\frac{1}{N}\sum_{n=1}^{N}[V_n \log \hat{V}_n + (1 - V_n) \log (1 - \hat{V}_n)]$$

$$FNCE = -\frac{1}{P}\sum_{p=1}^{P}[V_p \log \hat{V}_p + (1 - V_p) \log (1 - \hat{V}_p)]$$

Training Parameters

Train / Test Split = 0.9 / 0.1, Adam learning optimizer, Learning rate = 0.0003, Batch size = 60, Training Epochs = 500

Tech Stack

Tensorflow, cv2, Numpy , Pillow, Scipy, Python

Mobile App Details:

Here is the diagram showing how the mobile is integrated end-to-end from Users to Server and then back to Users.

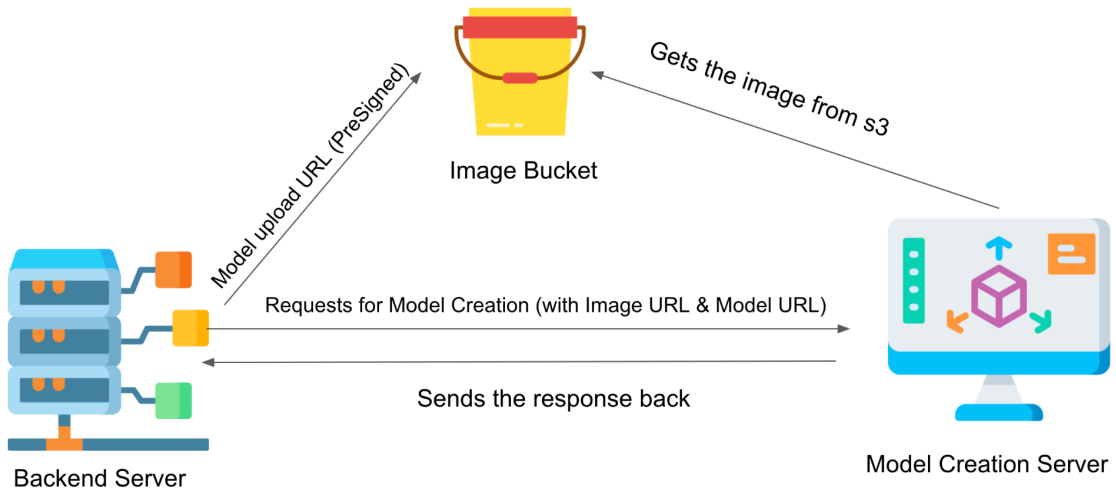


Figure 8 - Mobile App Interactions

AR Core

AR is a system that incorporates three basic features: a combination of real and virtual worlds, real-time interaction, and accurate 3D registration of virtual and real objects. The generated 3D model can be downloaded and placed in the real world after calibration of the motion and camera sensors with the surrounding. The placed model could be rotated from it's axis or zoomed in/out for perfect placement in the real world.

AR Core Functioning

ARCore uses three key capabilities to integrate virtual content with the real world as seen through your phone's camera:

- Motion tracking - ARCore detects visually distinct features in the captured camera image called feature points and uses these points to compute its change in location.
- Depth Understanding - ARCore creates depth maps, images that contain data about the distance between surfaces from a given point.
- Light estimation - ARCore can detect information about the lighting of its environment and provide you with the average intensity and color correction of a given camera image.

AR Core Functioning

Talking about glb/glTF conversion, glTF (GL Transmission Format) is a specification for the efficient transmission. glTF minimizes the size of 3D assets, and the runtime processing needed to unpack and use them. This format is supported by AR Core. We are using github to host our models. Talking about rendering in the App, the models can be downloaded and rendered in real time.

Model Retrieval using QR Code

Every Processed 3D model in our Database has a QR code associated with it, which contains information such as model name and the url to download the model.

Figure 9 - QR Example

{
"model_url":"https://btp-model.s3.ap-south-1.amazonaws.com/b74681da17f97068c83346dbf17b0902",
"title":"Table"
}

Figure 10 - QR Code output

QR Code Demo

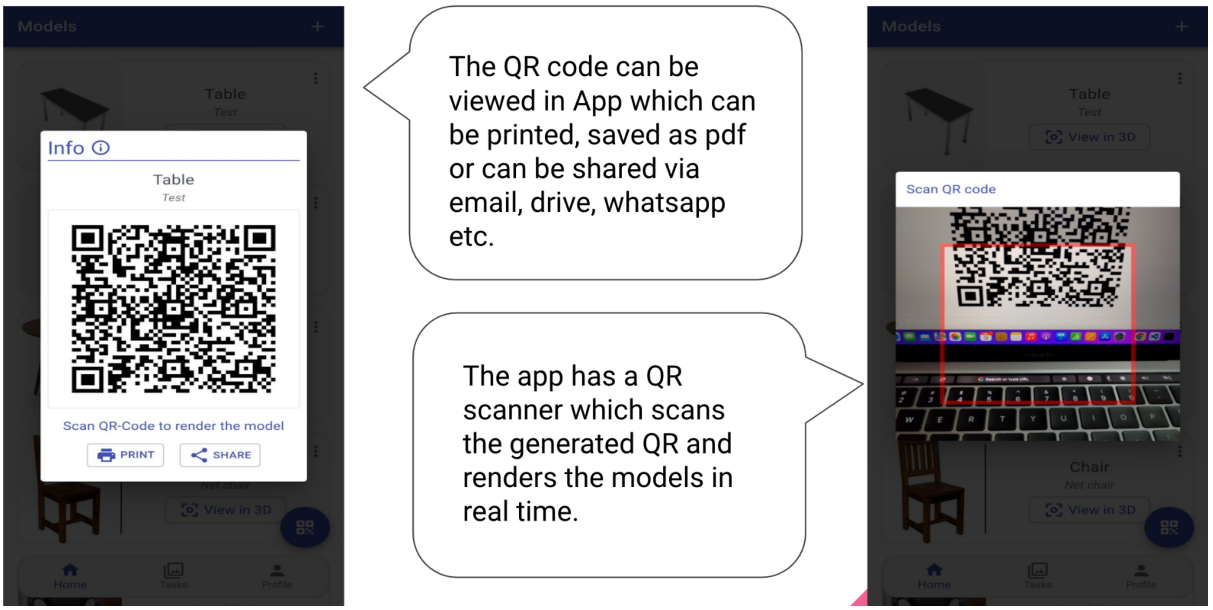


Figure 11 - QR Code Demo

Mobile App Tech Stack

ARCore, Android, NodeJS, ReactJS, AWS S3 Bucket

Major Challenges We Faced

1. High Computation Resources required for color learning

What is color learning?

For getting the colors to our 3D model from a 2D image, the idea is we build the same 3d structure. Each 3D voxel gets its shape from the corresponding voxel index. Two combined strategies - sample color from 2d image and other regress colors directly.

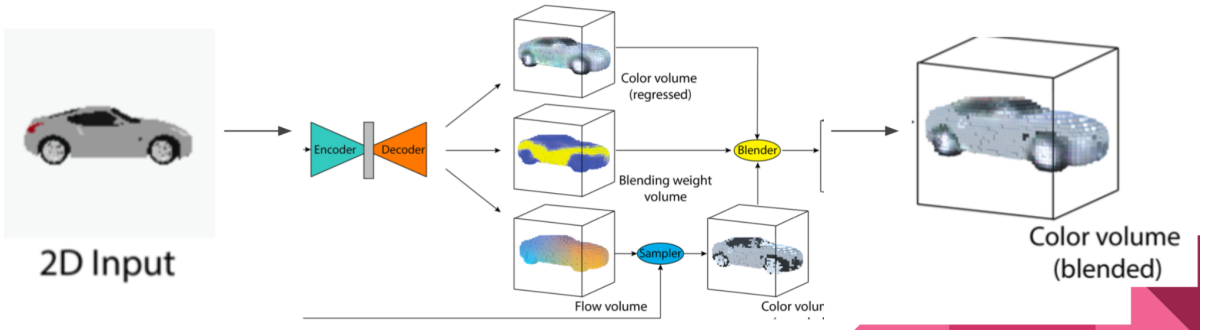


Figure 12 - Color Network pipeline

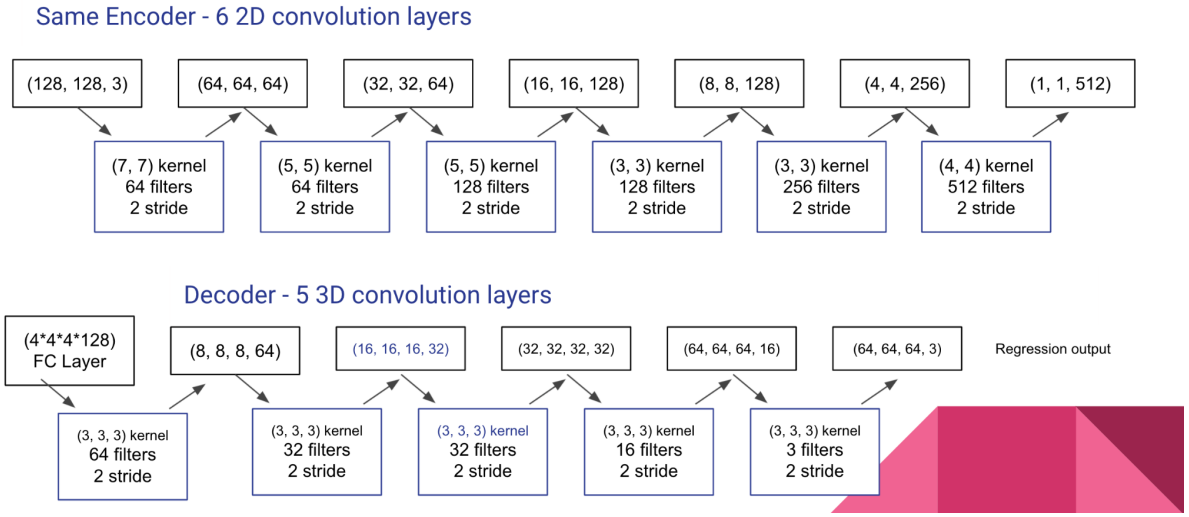


Figure 13 - Color Network semantics

Target flow loss	$L_{flow} = \frac{1}{S}\sum_{i=1}^{S} \|V_i^{flow} - \hat{V}_i^{flow}\|_2$	We only use Target flow during training.
Regression loss	$L_{clr\_regress} = \frac{1}{S}\sum_{i=1}^{S} \|V_i^{color} - \hat{V}_i^{clr\_regress}\|_2$	
Weighted loss	$\hat{V}_i^{clr\_blend} = w \times \hat{V}_i^{clr\_sample} + (1-w) \times \hat{V}_i^{clr\_regress}$	We only train for colored voxels.
Blend loss	$L_{blend} = \frac{1}{S}\sum_{i=1}^{S} \|V_i^{color} - \hat{V}_i^{clr\_blend}\|_2$	
FInal loss	$L = L_{flow} + L_{clr\_regress} + L_{blend}$	

Figure 14 - Color Network Loss functions

Complexity Explosion for color learning

- Number of images = 8.1k * 13 = 105k files
    - Only 10% extraction of files possible in drive in 13 hours
- Network * 4
- Model + First batch of data did not fit in google colab RAM.
- Training will dramatically slow down.
- We have the color learning architecture built up. We have documented the way to train the color learning if the interested person has enough compute resources to do so.

2. Finding unbiased metric to check model performance

- We need a metric to decide whether the model's performance is good or not in the furniture category.
- Looking at Mean Squared Loss is not a good choice, because for complex images even with higher loss, the resulting 3D structure can be visually appealing.
- If humans sit in the process of labeling good/bad outputs from the model, it adds that person's bias in the validation process.
- Finding a good unbiased metric for our use case, is an open ended research question. If time permitted, we wanted to go back to literature search and tackle this problem.
- This problem arise at the time of we collecting different distribution statistics of furniture category and checking the model performance on them.
- So, we only performed a handful of model experiments on out-of-distribution data.

RESULTS

Performance Analysis on Out-of-Distribution Data -

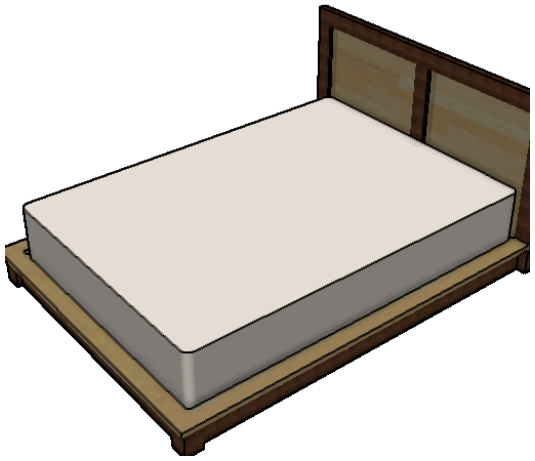
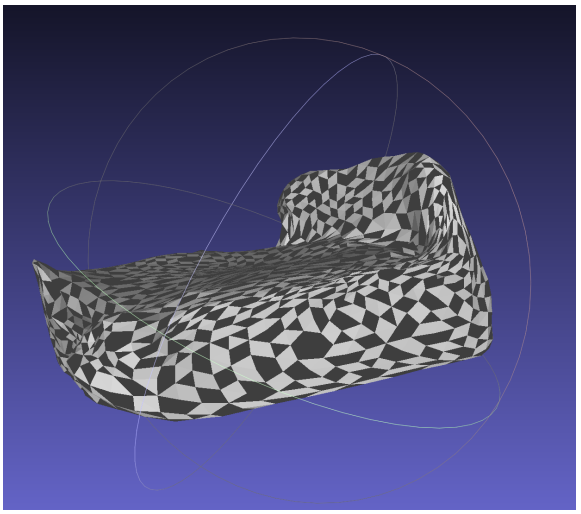
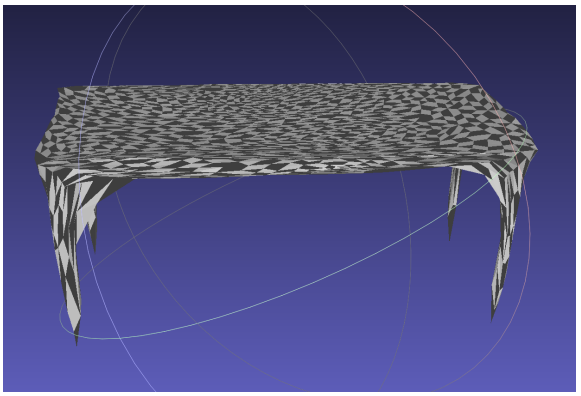
2D images	3D models
	

Table 1 - Result on Out-of-Distribution Data

We are getting visually appealing for our categories the model is trained on -

2D images	3D models
	

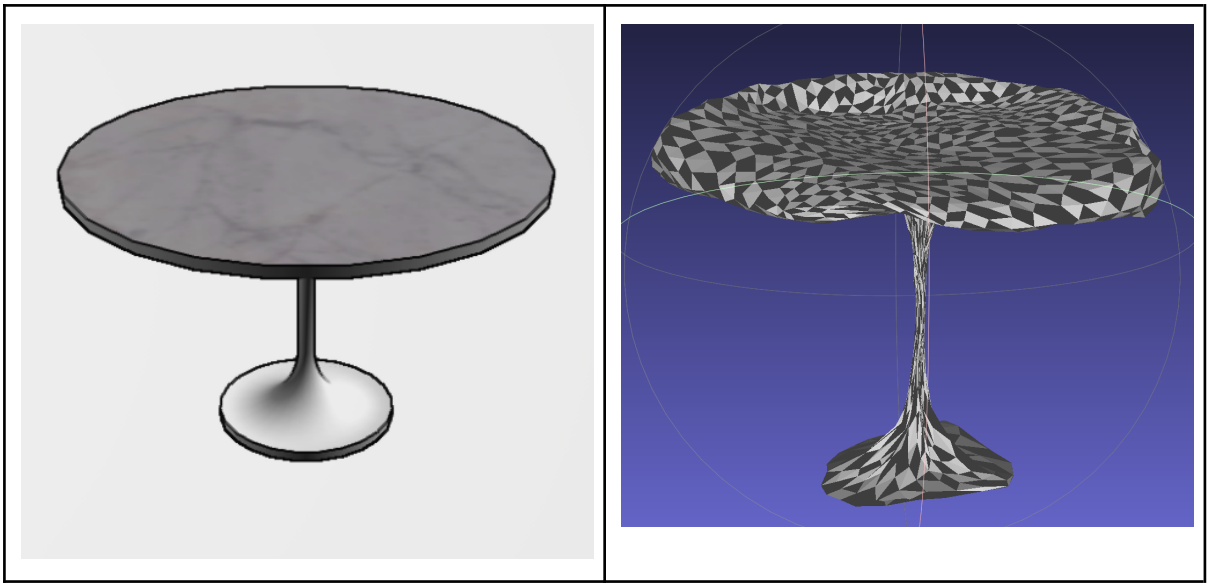


Table 2 - Results on In-Distribution Data

Figure 15 - Projecting 3D generated models in space

CONCLUSION

In this work, we built a novel end to end pipeline for shape recovery of 2d images for 3d model reconstruction. We made use of Parallel Encoder Decoder networks and sparse loss functions to achieve this. We got visually appealing results for furniture categories. We also successfully rendered 3d models in virtual space using AR Core. We also provided QR code base retrieval of 3d models. Our app provides functionality of zooming in/out, saving images and videos of projected 3d models. Whatever challenges we faced, we have extensively documented them. The work done here can be further carried out by solving the challenges. Finally, we want to thank our BTP advisor, Dr. Subu Kandaswamy for his guidance throughout BTP.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

1	AR	Augmented Reality
2	VR	Virtual Reality
3	LSTM	Long Short Term Memory
4	CAD	Computer Aided Design
5	GAN	Generative Adversarial Network

REFERENCES

- G. Pons-Moll, J. Romero, N. Mahmood, and M. J. Black. Dyna: A model of dynamic human shape in motion. *SIG- GRAPH*, 34(4):120:1–120:14, 2015 https://dl.acm.org/doi/10.1145/2766993

- M.Loper,N.Mahmood,J.Romero,G.Pons-Moll,andM.J. Black. Smpl: A skinned multi-person linear model. *TOG*, 34(6):248, 2015 https://smpl.is.tue.mpg.de/

- T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros. View synthesis by appearance flow. In *ECCV*, pages 286– 301, 2016 https://arxiv.org/abs/1605.03557

- C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d- r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, pages 628–644, 2016 https://arxiv.org/abs/1604.00449

- R. Zhu, H. K. Galoogahi, C. Wang, and S. Lucey. Rethinking reprojection: Closing the loop for pose-aware shape recon- struction from a single image. In *ICCV*, pages 57–65. IEEE Computer Society, 2017 https://arxiv.org/abs/1604.00449

ACKNOWLEDGEMENTS