Voxel-based 3D Detection and Reconstruction of Multiple Objects from a Single Image

Md Ragib Shaharear, Darsh Kishorkumar Shah, Devang Pravinbhai Kakadiya, Rikenkumar Patel
Ira A. Fulton Schools of Engineering,
Arizona State University, Tempe, USA.
mshahare@asu.edu, dshah58@asu.edu, dkakadiy@asu.edu, rpatel80@asu.edu

***Abstract*—A long-standing goal of computer vision has been to infer 3D locations and forms of many objects from a single 2D image. Most existing research either predicts one of these three-dimensional features or attempts to solve both for a single item. One of the most challenging problems is learning an effective image representation that is well-suited for 3D detection and reconstruction. We suggest that using a 3D feature lifting operator, we can learn a regular grid of 3D voxel features from an input image that is aligned with 3D space[1]. Our mobilenetSSD and YOLO detection formulates 3D detection in 3D space, based on 3D voxel characteristics. In addition, we create an efficient coarse-to-fine reconstruction module that includes coarse-level voxelization and a unique representation, allowing for fine detail reconstruction and one-order-of-magnitude faster inference than previous approaches. With the use of complementing supervision from both 3D detection and reconstruction, the geometry and context of 3D voxel features can be preserved, which benefits both tasks. 3D detection and reconstruction in single and multiple object scenarios highlight the usefulness of our approach.**

***Index Terms*— 3D Object Detection, mobilenetSSD and YOLO detection, 3D Shape Reconstruction, Generic Object 3D Reconstruction.**

I. Introduction

Due to its importance in Robotics, AR/VR, and autonomous driving, one key difficulty in machine learning is learning an acceptable representation of the image that is suited for 3D recognition and reconstruction. It's challenging to correctly recreate the scene context (both geometry and semantics) from a single snapshot due to the complexity of real-world circumstances and the wide range of category variations. A coarse-to-fine reconstruction module is introduced, which includes coarse-level voxelization and representation. This enables adequate detail reconstruction and inference that is one order of magnitude faster than previous methodologies[2].

As a consequence, attempts like grid-based representation for tasks like rendering, detection, and reconstruction have been made. OFT proposes that image features be sampled and transformed into a BEV grid representation, allowing for comprehensive reasoning of the 3D space configuration. CaDDN adds a categorical depth prior to the BEV grid representation, resulting in improved 3D detection accuracy. DeepVoxels and UCLID-Net construct voxel features by projecting 2D features into 3D space for rendering or single-object reconstruction. For predicting 3D bounding boxes and surfaces of multiple objects from a single image, we propose a novel voxel-based 3D detection and reconstruction framework(see fig1).

Figure 1: 3D boudning box around object

To be more specific, First, we divide a 3D scene space into a regular grid of voxels. We assign 3D features to each voxel by sampling from the image plane with a 2D-to-3D feature lifting operator and the known camera projection matrix. Because multiple voxels can be projected to the same position, similar features appear along the camera ray, increasing the difficulty of downstream tasks. To discuss this, we implement a positional encoding strategy to improve the discrimination and position-awareness of our voxel features. We carefully design our detection and reconstruction modules based on intermediate voxel features.

We provide an end-to-end multi-level framework that infers 3D geometry objects in a pixel aligned way at an unprecedentedly high resolution, preserving the features in the original inputs without any post-processing. In contrast to coarse-to-fine techniques, our method does not impose an precise geometric representation in the coarse levels.[3] Rather, implicitly encoded geometrical context is conveyed to higher levels without making an explicit geometry determination too soon. Our technique is based on the Pixel-Aligned Implicit Function (PIFu) model, which was recently introduced. The representation's pixel-aligned structure allows us to smoothly combine the holistic embedding learnt by coarse reasoning with picture features learned from high-resolution input in a principled manner.

Eventually, the algorithm must recover the backside, which is not visible in any single image, in order to complete the reconstruction. Missing information that is not anticipated from observable measures, such with poor resolution input, will result in too smooth and hazy estimates. Conditioning our multi-level pixel-aligned form inference with the inferred back-side surface normal eliminates ambiguity and improves visual quality by ensuring that the viewable and occluded regions have the same amount of detail.[4]

As we mentioned in datasets, On the training and validation data sets from PASCAL VOC 2007 and 2012 we train the network for about 135 epochs. Also, when testing on 2012 we also include the VOC 2007 test data set for training.

II. Method

A. Object Detection

First, we used the CenterNet-3D algorithm to detect the objects, but we found out it has fewer categories, so we explored other algorithms to detect objects and found YOLOv3. YOLO(You Only Look Once) is an extremely fast multi-object detection algorithm that uses a convolutional neural network (CNN) to detect and identify objects.[5] YOLO is a real-time object detection approach with a wide variety of applications in computer vision. To detect elements such as height, breadth, center, and object classes, this method use a single bounding box regression. [6]

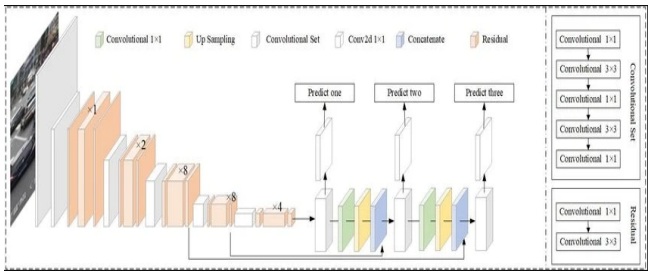

Figure 2: The Architecture of YOLO

There are 53 convolutional layers in our detection network, followed by two fully linked layers. The feature space from prior layers is reduced by alternating 1 1 convolutional layers. There are 53 convolutional layers in all, including a batch normalization layer and Leaky ReLU activation. The input image was 416 by 416 pixels in size. The end result is a list of bounding boxes together with the classes that were discovered. Each bounding box is represented by six digits (pc, bx, by, bh, bw, c). Box weight, height, and coordinates are represented by bx, by, bh, and bw. C is a collection of classes. The probability of that class is Pc.

Common Objects in Context (COCO) is a database that aims to make future research in object detection, instance segmentation, picture captions, and human key-point localization easier. COCO is a massive dataset that comprises large-scale object detection, segmentation, and captioning.

YOLOv3, in particular, has three anchors. This results in the prediction of three bounding boxes per cell. Objects can sometimes be detected by multiple bounding boxes. To solve this problem, we used Non-max suppression. It only passes the detection which is not previously been detected.

YOLO-based models do not attempt to take over the world, and their operation is based on three basic techniques:

1) Residual blocks: The model divides the input image into grids of equal size, each of which is responsible for detecting an object or a fragment of an object that appears within the grid.

2) Bounding box regression: Each cell has a bounding box with attributes that highlight things within it, such as weight, height, class, and center. YOLO predicts these using a bounding box regression, which indicates the chance of an object appearing in the bounding box.

3) Intersection over Union (IoU): IoU describes the overlap of bounding boxes. Each grid cell predicts the bounding boxes and their confidence scores. By dividing the overlap area by the union area, the IoU is determined. The IoU is 1 if the predicted bounding box and the ground-truth bounding box are identical. It's much easier to remove bounding boxes that are too different from the genuine box.

For each spatial cell in the output layer, YOLOv3 predicts numerous boxes. For YOLOv3, that number is 3. Anchors are the boxes that are centered in the cell.

The YOLO loss for every bounding box prediction are based on the following terms.
1. Coordinate loss - object does not cover while predicition
2. Objectness loss - if box-object IoU prediction is an incorrect
3. Classification loss - This occurs as a result of the object in that box failing to forecast '1' for the correct classes and '0' for all other classes.

$$\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

Here, xi and $\bar{x}i$ is the coordinate of x axis for ground and predicting bounding boxes respectively. Also, yi and yi is the coordinate of y axis for ground and predicting bounding boxes respectively, same apply of wi,hi,ci and pi(c). 1iobj denotes if object appears in cell i and 1ijobj denotes that the jth bounding box predictor in cell i is responsible for the prediction. A residual scale prediction was used to replace the second line. As a result, the loss was proportional to relative scale error rather than absolute scale error.

YOLOv3 creates a 2D bounding box around the object. To create a 3D bounding box, we have used mediapipe algorithm. It was trained on the objectron dataset, which has around 15000 annotated videos and 4M annotated images. It has object-centric multi-views, observing the same object from different angles.

There are two type in mediapipe. First stage pipeline and second stage pipeline.

The model backbone, which is built on MobileNetv2, has an encoder-decoder architecture, as shown in Fig 3. We use a multi-task learning strategy, combining detection and regression to estimate the form of an item. The form task predicts the shape signals of an object based on existing ground truth annotations, such as segmentation. If there is no shape annotation in the train-

ing data, this is not required. We utilize the annotated bounding boxes for the detection task and fit a Gaussian to them, with the center at the box centroid and standard deviations proportional to the box size. The purpose of detection is to forecast the peak of this distribution, which represents the object's center location. The regression job calculates the 8 bounding box vertices' 2D projections. We use a well-known posture estimation approach to get the final 3D coordinates for the bounding box (EPnP). It can determine an object's 3D bounding box without knowing its dimensions beforehand. We can simply compute the object's posture and size using the 3D bounding box.

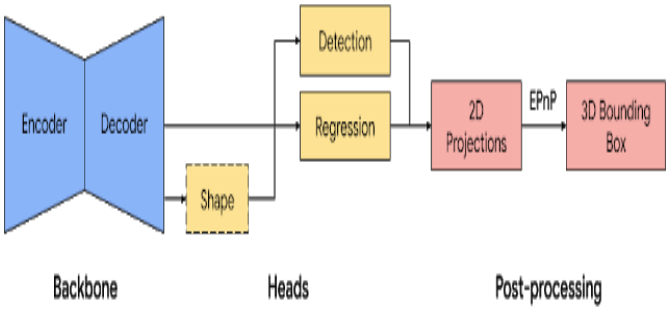

Figure 3: Network architecture for single-stage 3D object detection

To construct a 3D bounding box, we employed a two-stage workflow. There are two steps to the two-stage pipeline. The first stage employs an object detector to locate the item's 2D crop. The picture crop is used in the second stage to estimate the 3D bounding box. At the same time, it computes the object's 2D crop for the following frame, eliminating the need for the object detector to execute every frame.

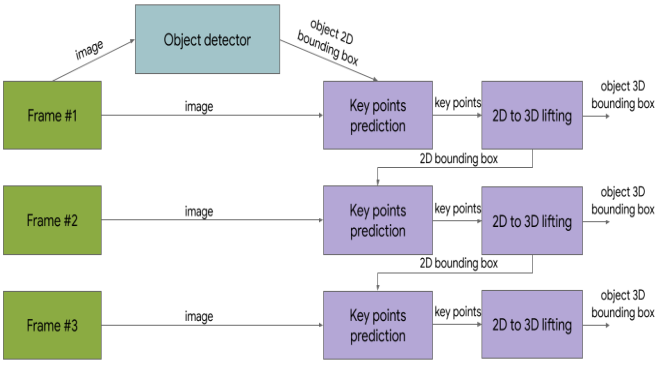

Figure 4: Network architecture for two-stage 3D object detection

B. 3D Reconstruction

We provide an end-to-end multi-level system that infers 3D reconstruction in a pixel aligned manner at an unprecedentedly high 1k image resolution, preserving the features in the original inputs without any post-processing. In contrast to coarse-to-fine techniques, our method enforces no explicit geometric representation in the coarse levels. Rather, higher layers receive implicitly encoded geometrical context without making an explicit geometry determination too soon. The Pixel-Aligned Implicit Function (PIFu) model and the Front to Back Inference model are used in our approach. The representation's pixel-aligned structure allows us to use principled reasoning to merge the holistic embedding obtained through coarse reasoning with picture features learned from high-resolution input. Each level gradually adds new information that was absent in previous stages, with the ultimate geometry conclusion being completed only at the highest level.

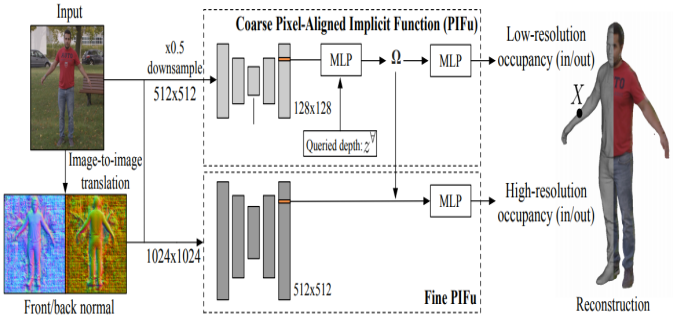

Figure 5: Overview of framework

Pixel-Aligned Implicit Function :

In PIFu we used neural implicit function for shape representation, 3D shape are represented by the label set of occupancy fields parameterize by a neural network. Since formulation does not any discretization, we can effectively model 3D shape at an arbitrary resolution. All the work including occupancy network and encode an input image as a global descriptor to reconstruct corresponding 3D shapes. However, this way the network does not leverage special relationship between corelating 3D points and the input image. To address this issue, we introduced pixel aligned implicit function where we utilize free convolutional image features to associates queries 3D points and the projected pixel coordinate given the pixel align the code and query to depth value Z we can inverse 3D occupancy fields in a pixel aligned manner. However we found that the reconstruction quality is still bounded by the feature resolution of an image encoder, the original pie for implementation utilize an image encoder that has sampling ovulation to keep the spatial resolution small for the following reason: First small feature resolution allows for istic reasoning as 3D reconstruction requires absolute to location aware the surface holistic reasoning is indispensable over small resolution allows us to train the model with a reasonable amount of time.

The downside is limiting the expressiveness up to the small resolution resulting in less details construction than the original input image. What about utilizing a shallow but high-resolution image encoder, although the result becomes sharper and more detailed the reconstruction is pawn to fact due to the lack of global information. In this work we addressed the disadvantages of both architectures by a coarse to fine multi-level approach, there are several 3D construction methods that utilize a cost to find strategy however we argue that since the existing methods have several limitations the existing approaches also elaborates a memory efficiency from the cost model. PIFu's purpose is to create a Implicit function, f(X;I), that predicts the binary

occupancy value for every given 3D position.

f(X; I) = 0 € R

where X is the 3D position in continuous camera space and I is the single image.

Front-to-Back Inference :

Finally, we need Front to Back Inference for a complete reconstruction. The system must retrieve the backside of the object, which is not visible in any image. It takes predicted frontside and backside normal maps using the following equation.

$f(X) = g^L \phi^L(x^L; I^L; F^L; B^L;); Z$

where I the is input and F and B are predicted normal maps at the same resolution. is the image feature extractor , X is the projected 2d location.

Our approach is based on the Pixelaligned Implicit Function (PIFu) architecture, which accepts images with a resolution of (512x512) as input and produces low-resolution feature embeddings (128x128). The fine module receives higher resolution images (1024x1024) as input and encodes them into high-resolution image characteristics (512x512). The second module uses the high-resolution feature embedding as well as the 3D embeddings from the first to forecast an occupancy probability field. To improve the quality and realism of the reconstruction, we first estimate normal maps for the front and back sides in image space and feed these to the network as additional input.

III. Implementation and Simulation

We created code to find single and numerous items from a single image in this project. We have implemented the YOLO algorithm to detect an object. We found out that it gives better accuracy than MobileNetSSD. YOLO has Real-time frames processing at 45 fps. It is less false positive in the background. It has higher detection accuracy. To train YOLO, we have used the COCO dataset. The picture collection was built to improve image identification. Therefore COCO stands for Common Objects in Context. The COCO dataset offers demanding, high-quality visual datasets for computer vision, with the majority of the datasets containing state-of-the-art neural networks. The COCO dataset has 80 images. We have successfully run the code for 3D detection.

For the 2D bounding box, we have used YOLOv3. We download some images from the internet to test our code. Images were detected accurately. Now for the 3D bounding box, we have used the Mediapipe algorithm, which takes input as an image and gives a 3d bounding box around the object as an output.

The next part is 3D Construction. we were planning to form a 3D line drawing so we wanted the Bresenham algorithm. In the Bresenham algorithm, we have to give two 3D coordinates, and from that, we can find the line joining them. The point clouds used as input for the reconstruction procedure were from both real range scans and synthetic datasets. 3D reconstruction was divided into three parts. 1) Voxelization and gap-filling, 2) topological thinning, 3) meshing. But due to unavailability of programming code, we have planned to go with PIFU algorithm.

For 3D reconstruction, our goal was to create all the objects from the image but initially it was difficult task for us as well as our system so we decided to go with individual image reconstruction. As we were able to detect all the objects from a single image, we performed image segmentation to segregate all the objects and present them individually. As there is single object image, its now achievable task for us for reconstruction. we Used PIFu algorithm for Reconstruction human images as that is stat-of-art algorithm. PIFU is trained with human poses and key feature of human poses so by uploading human images, it estimate the depth of front and back pose with high accuracy and gives back the 3D human model with pose. the PIFU task we performed uses single level Pixel-Aligned implicit function to estimate pose and reconstruct the object.

We're also evaluating additional algorithms, such as PIFuHD, which creates an end-to-end trainable multi-level architecture. A coarse level looks at the entire image at a lower resolution and concentrates on holistic thinking. This gives context to a fine level that uses higher-resolution photos to estimate very detailed geometry.[8] The challenging task in reconstruction is the background inference, that causes poor object reconstruction. PIFu model uses COCO detaset to test with different backgrounds to reduce the need of image segmentation so we don't have to consider that as a pre-process.

We're were working on a Windows 10 ACER system with an Intel(R) Core (TM) i5-6200U processor operating at 2.30GHz-2.40GHz, 12.0 GB RAM, and an NVIDIA 940 MX graphics card. now, we have upgraded to a Lenovo system with Windows 11 and an AMD Ryzen 7- 5800H Processor (3.20 GHz, up to 4.40 GHz Max Boost, 8 Cores, 16 Threads, 16 MB Cache) with 16GB RAM and an NVIDIA® GeForce® RTXTM 3060 6GB Graphics card for faster performance and larger datasets, as we were planning to use 19 objects detection instead of the previous nine object detection, but as we are implementing the YOLO model for 3D detection models we have 80 different objects/classes that we can detect.

We are using Visual Studio Code(version 1.65) for the platform, which can be downloaded from https://code.visualstudio.com/ to run the python code with python(version 3.10.3), which is available on https://www.python.org/downloads/. This two software are the simple procedure for installation as selecting the drive and agreeing the license terms, but the main setup comes after the initial installation.

We need to download and install various library packages for Python to operate in Visual Studio Code. Once Python and Visual Studio Code have been installed, the newest PIP package must be installed first. We're using Pip version 22.0.4, which you can get by entering "pip install pip" at the command prompt (VS Code). For 3D Object detection, we ran the code with google Collaboratory for ease, but eventually, we established a fully performing environment in Visual Studio Code. For 3d reconstruction, we will use conda to train the model. The github file download from

IV. Result

The experiment is split up into two parts, the first of which is object detection and the second of which is 3D model reconstruction. We were using various neural network models

in both segments to compare the model's accuracy (which we call "confidence") on a specific image objects and how many object objects it can recognize from a single image.

A. Object Detection from an Image:

At the outset, we introduced the scanNet3D model and the mobileNetSSD model for object detection. We took into account the input image, which included various areas of the house such as the kitchen, bedroom, living room, and study room, among others. We found from the initial experiment that considering network models have difficulty detecting objects from input images (even after lowering the confidence), and that the live stream video has too much glitch and detection inaccuracy (detecting chair as sofa and mobile as TV monitor).[9]

Further research revealed that YOLO is the state-of-the-art neural network for image detection when using models with 2D and 3D bounding boxes, and that the detection accuracy is deemed to be quite high when compared to the prior models we utilized hence, we have decided to implement the YOLO model for our project. Here are some examples of detection results from both models, including both detection and object outcomes.

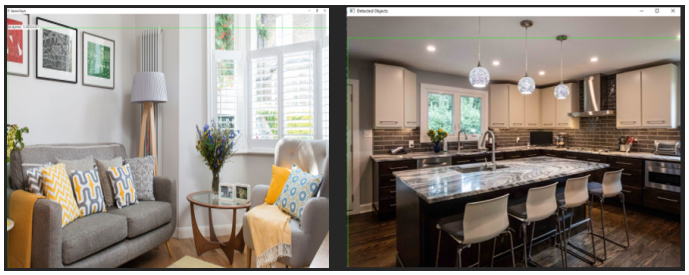

Figure 6: Results from mobileNetSSD:

The image shows results of living room and kitchen images detecting the whole image as an object along with that the class in the kitchen is missing as well as class in living room is inaccurate.

Figure 7: Results from YOLO:

The image shows results of living room and kitchen images detects the objects with almost full accuracy and the object labelling are also accurate compared to the other models.

B. 3D reconstruction of objects from images:

We explored 3D reconstruction of the observed objects for the project's second section. We used the software such as "blender" or "Mayavi" to reconstruct the thing into a three-dimensional shape and portray it.

We performed the reconstruction working on the coding for this section of the project, in which we proposed to divide the voxel around the objects into equal-sized little boxes, each box containing the coordinate values.[10][11] We save all of the values in a text file that we use as the input for a 3D reconstruction model, and then we represented those coordinates in blender (the method we're currently working on) and Mayavi (the model which we have output but still not accurate for some objects).Unfortunately, we have decided to go with blender rather than mayavi because, it gives better object representation as the 3D rendered object. This model was already pre-trained that we used directly for testing for our images.

Here is the current result we are having for the detecting of a single person(brother) and reconstruct it into 3D model in blender.

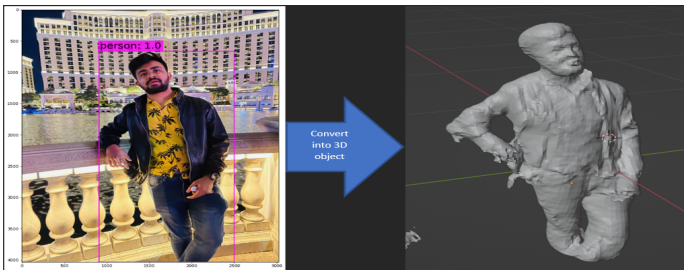

Figure 8: the 2D image detection with YOLO model to 3D model in blender.

Figure 9: Different angles of the 3D model created with PIFu in blender.

V. Conclusion

This paper presents 3D detection and reconstruction. We have developed a method for detection and reconstruction after going through many methods and choose the best method to get the accurate 3D reconstruction,. We have tried the code that the author has presented which turned out to be partially available so we have combined two different approaches. Apart from that, we also researched object detection using a different algorithm and compared it with the author's code.[12] We found out that YOLO provides better accuracy and can be trained using various labels, and for 3D reconstruction, we are still in the process. We re-developed code for 3D reconstruction, but we will also try different algorithms for that and select the model that gives a better result.

Our team has conducted numerous studies on object detection and 3D reconstruction throughout the project. In addition, the issue statement we identified and the code we chose to work

on were both incomplete. That allowed us the opportunity to experiment with and work on various neural network models, which expanded our understanding and eventually led to the notion of running the detection more rapidly and easily.

The possible issue we discovered was that we did not devote enough time to selecting useful code that could assist us at the outset, so that in the future, we can focus a little more on selecting supporting documents that will make our entire process easier and result in fewer conflicts throughout the project.

VI. Discussion and Future works

Considering the simple object detection and reconstruction, we have explored many methods for both tasks.As we had incomplete code at first, we went thought and finish it but find out that it doesn't work as we expected. Eventually we adapted different network model that satisfied our need for object detection and we moved to our next phase of he project. For 3D reconstruction, we have conducted many methods from PCA, SDF to PIFu, where we found out that representation of the detected object was quite challenging task for us. Also no one from our team had and relative experience or educational background for perception and reconstruction so this project gave us all an opportunity to learn something new as well as work with different neural networks from training to validation and at the end, test. During the whole process, we figured that all the thing we decided, didn't go as we planned but after the results we received, it certainly gives us the idea how to work with neural networks and we are hoping to keep this excitement in the future and complete our future expectation for this project. Also during the whole process, we learned how to handled all the different task of the project so if we get an opportunity to work on the large scale project, we will have an idea how to deal with all the different segment of that. We have also figured that our system requirement was comparatively low for training the neural network and as the result we have used pre-trained model for 3D-reconstruction so in future, we are hoping to work with AGAVE super computer in order to work with training of our model.

References

[1] Liu, Feng, and Xiaoming Liu. "Voxel-based 3D Detection and Reconstruction of Multiple Objects from a Single Image." Advances in Neural Information Processing Systems 34 (2021).

[2] Shunsuke Saito, Tomas Simon, Jason M. Saragih, Hanbyul Joo: PIFuHD: Multi-Level Pixel-Aligned Implicit Function for High-Resolution 3D Human Digitization. CoRR abs/2004.00452 (2020)

[3] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, Alexander C. Berg:SSD: Single Shot MultiBox Detector. CoRR abs/1512.02325 (2015)

[4] Y.C. Chiu, C.Y. Tsai, M.D. Ruan, G.Y. Shen and T.T. Lee, "Mobilenet-SSDv2: An Improved Object Detection Model for Embedded Systems," 2020 International Conference on System Science and Engineering (ICSSE), 2020, pp. 1-5, doi: 10.1109/ICSSE50014.2020.9219319.

[5] Joseph Redmon, Ali Farhadi:YOLO9000: Better, Faster, Stronger. CoRR abs/1612.08242 (2016))

[6] Xin Huang, Xinxin Wang, Wenyu Lv, Xiaying Bai, Xiang Long, Kaipeng Deng, Qingqing Dang, Shumin Han, Qiwen Liu, Xiaoguang Hu, Dianhai Yu, Yanjun Ma, Osamu Yoshie: PP-YOLOv2: A Practical Object Detector. CoRR abs/2104.10419 (2021)

[7] Chen-Hsuan Lin, Chaoyang Wang, Simon Lucey: SDF-SRN: Learning Signed Distance 3D Object Reconstruction from Static Images. CoRR abs/2010.10505 (2020)

[8] Slavcheva, Miroslava Kehl, Wadim Navab, Nassir Ilic, Slobodan. (2016). SDF-2-SDF: Highly Accurate 3D Object Reconstruction. 9905. 680-696. 10.1007/978-3-319-46448-$0_4$1.

[9] Shichen Liu, Tianye Li, Weikai Chen, Hao Li: Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning. CoRR abs/1904.01786 (2019)

[10] Haozhe Xie, Hongxun Yao, Shengping Zhang, Shangchen Zhou, Wenxiu Sun: Pix2Vox++: Multi-scale Context-aware 3D Object Reconstruction from Single and Multiple Images. CoRR abs/2006.12250 (2020)

[11] Haoqiang Fan, Hao Su, Leonidas J. Guibas: A Point Set Generation Network for 3D Object Reconstruction from a Single Image. CoRR abs/1612.00603 (2016)

[12] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, Silvio Savarese: 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. CoRR abs/1604.00449 (2016)

This table is required and to be included in the final report
(this table does not contribute to the page count)
Please don't change the format of the table, i.e., only fill in the blanks

end of semester reflection - lessons learned from working on the final project

Team # and names of team members

Team# 9 **Rikenkumar Patel** **Darsh Shah** **Devang kakadiya** **Ragib Shaharear**

Project title **Voxel-based 3D Detection and Reconstruction of Multiple Objects from a Single Image**

	literature (not well written or self-contained, not specific on implementation, no data source	**setting up the environment and obtaining data**	**to have the first successful test run (issues during**	**obtaining results (algorithm/method is dificult to implement,**	**obtaining results (cannot duplicate what was reported**	**reporting (Intro, method, result, discussions, …)**
specific & detailed evidence is required to support claims (e.g., links, repository sites, equation #, figure #,	The initial literature was well written and explained everything but the source code was not complete. We have also tried to contact the author multiple times but we havent received any responses.	The initial code we picked was not complete. We setup the enironmnt and completed the code but not expected result but	as we havent recived our first code running properly after the results, we planned to go with different state-of-art model for our project.	at first, out system was not competible for training so we have updated out system. For detection, we were able to make code and train the model and test it but	As we differed the model, we havent duplicate the result but eventually we end up duplicating and even getting better result out of the model we used.	reporting was little easy task for us as compared to other but as we have conducted multiple researches, we

footnote (e.g., reference citation…)

every team member is required to complete this form, which will be reviewed and signed by other team members
by signing this form, you agree that the student's descriptions of contributions are factual
effort/contribution levels determine an individual's score which can be different from the rest of the team
This form does not contribute to page count.

Assignment name/key words Final Project: Voxel-based 3D Detection and Reconstruction of Multiple Objects from a Single Image
Team # Team 9

Student name: Rikenkumar Patel 25%	worked on literature	worked on implementation (data, platform, test run, debug, compatibility…)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, …)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragraphs, equation #, figure #, code line #'s sections, etc.…)	found and explored the research articles such as: https://doi.org/10.48550/arXiv.2111.03098, https://doi.org/10.48550/arXiv.2004.00452	Work on the code for 3D reconstruction and testing of the model.	created dataset for testing collecting the images from the internet and surrounding.	worked on the report for implementation of 3D reconstruction and results. Also worked on lesson learned part of the report.	Helped in starting writing paper in overleaf, table preparation, giving idea to form the structure of writing.	Darsh Shah	Devang kakadiya	Ragib Shaharear

Student name: Darsh Shah 25%	worked on literature	worked on implementation (data, platform, test run, debug, compatibility…)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, …)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragraphs, equation #, figure #, code line #'s sections, etc.…)	found and explored the research articles such as: https://doi.org/10.48550/arXiv.1512.02325, https://doi.org/10.48550/arXiv.2110.06922	Working on the code for object detection for YOLOv3 and mediapipe.	worked on the mediapipe and yolov3 models for validating and testing for accuracy.	worked on the report for the method and implementation part for object detection of multiple objects.	worked on setting the group meetings and correcting the proper representation of equations and tables.	Rikenkumar Patel	Devang kakadiya	Ragib Shaharear

Student name: Devang Kakadiya 25%	worked on literature	worked on implementation (data, platform, test run, debug, compatibility…)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, …)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragraphs, equation #, figure #, code line #'s sections, etc.…)	found and explored the research articles such as: https://doi.org/10.48550/arXiv.2104.10419, https://doi.org/10.48550/arXiv.1904.01786	working on the environment setup for the experiment and compatibility issue with our system.	worked on setting up the system for better runtime and results. Conducted results in to better representation for ease of the project.	worked on the abstract, introduction and conclusion part of the final report. Also helped working on team contribution form.	Helped in Proofreading the report and correct any possible mistakes.	Rikenkumar Patel	Darsh Shah	Ragib Shaharear

Student name: Md. Ragib Shaharear 25%	worked on literature	worked on implementation (data, platform, test run, debug, compatibility…)	generated results (run results, result data processing, presenting results	wrote report (Intro, method, result, discussions, …)	other significant contributions	peer approval 1	peer approval 2	peer approval 3
specific & detailed evidence is required to support claims of contributions (make reference to specific paragraphs, equation #, figure #, code line #'s sections, etc.…)	found and explored the research articles such as: https://doi.org/10.48550/arXiv.2010.10505, https://doi.org/10.48550/arXiv.2006.12250	Working on the training of the models we used for object detection.	worked on collecting data for the training and processing the images. Setting up and dividing the dataset into train, test and validate format.	worked on the method part of the 3D reconstruction explaining the PIFu model and helped with references of the final report.	Helped in arrange various sections into one report and setting the report in the correct IEEE report format.	Rikenkumar Patel	Darsh Shah	Devang kakadiya