International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 12 | Dec 2020 www.irjet.net p-ISSN: 2395-0072

Automated 2D Image to 3D Model Construction: A Survey

Mahipal Mehta[1], Shweta Kothawade[2], Sanket Kudale[3], Sukhada Dole[4]

[1,2,3,4]*UG Students, Department of Computer Engineering, Marathwada Mitra Mandal's College of Engineering, SPPU, Pune, India.*

Abstract - *Rapid growth in computer graphics to enhance user experience in almost every sector like e-commerce, education, healthcare, gaming; calls for 3D representation unlike 2D representations a few years back. Applications like e-commerce need 3D models of real objects rather than a generalized object. This brings a massive demand for 3D model reconstruction from a 2D image. Till now, the 3D models are developed by graphics designers using various graphics software like Blender, MAYA and Unity3D, etc. But this procedure is very tedious, time-consuming, and manual. The surveyed techniques extend remarkable state-of-art work in 2D recognition and try to push it into 3rd dimension. Our paper explores some techniques based on the comprehensive survey of different research papers that help to convert 2D images into 3D models using deep learning techniques including Mask-RCNN, graph convolutions, image encoders and decoders using convolutions, Pytorch3D's open-source library. Our paper acknowledges limitations in 3D modeling with 3D supervision and how it can be solved using 2D supervision handling the tradeoffs between resources and the accuracy of the constructed models.*

Key Words: **Mask RCNN, Instance Segmentation, Graph Convolution Network, Differential Rendering, Unpooling**

1. INTRODUCTION

In recent years, deep learning has demonstrated outstanding capabilities in solving 2D-image tasks such as image classification, object detection, semantic segmentation, instance segmentation, and human pose estimation. There are some engineering challenges involved in deep learning with respect to 3D modeling (3D Deep Learning) [5]. As we know, 2D images are almost universally represented by regular pixel grids. But in contrast, 3D data are stored in various data structures as voxel grids, point clouds, and meshes which can exhibit heterogeneity which is one of the major challenges when dealing with 3D deep learning techniques [5].

Voxels(Volumetric Pixels) in 3D are same as Pixels in 2D. Voxels generally divide 3D space into uniform 3D cells, typically cubes, which is known as voxel grid. 3D meshes or polygon meshes consist of a combination of vertices, edges and faces [7]. A point cloud is just a set of 3D data points and each point is represented by three coordinates in a coordinate system like Cartesian [7]. Each of these representations has its own benefits and drawbacks concerning accuracy, computational power, memory requirements, and representational details. Therefore, the choice of data representation directly affects the approach that can be utilized to apply 3D deep learning [4].

To work with 3D deep learning some techniques are built on 3D supervision. A renderer takes input as scene information (textures, lights, materials, camera, etc) and outputs an image [5]. Differentiable renderer allows the gradients with respect to loss of predicted objects and ground truths, to be calculated and backpropagated through their respective 2D images [8]. This drops the need for 3D supervision of data [8]. We should endeavor to build a system that can operate on unconstrained real-world images with many objects, occlusion, semantics, and diverse lighting or shading conditions but that should not ignore the rich 3D nature of the ground truth [4].

The process of conversion from a 2D image to a 3D object includes different steps. Initially, the object to be reconstructed needs to be segmented from its background. Instance segmentation which is achieved using techniques such as Mask-RCNN is used to segment the exact shape of the object from a 2D image. Predicting the 3D shape of the object using 3D deep learning is followed by a differential rendering process. Differential rendering tries to refine the predicted 3D shape with respect to the ground truth by calculating losses and back-propagating using gradients. The rendering process can be 2D supervised or 3D supervised. This step is crucial in deciding the success of the technique with considerable parameters.

2. RELATED WORK

The paper includes scrutiny of different surveyed techniques mentioned in different papers which will help to analyse and to implement the accurate, efficient method to convert 2D images into 3D models.

2.1 Autosweep: Recovering 3d editable objects from a single photograph [1]

[1] have introduced a fully automatic novel method named AutoSweep that can recover 3D Editable Objects from a single RGB image. This paper proposes a technique that focuses on recovering 3D objects with semantic parts and which can be directly edited. The proposed technique is based on '3-Sweep'[6]. The proposed method not only recovers the shape of the objects but also the surface texture and colors. It assumes the objects present in the image build from two types of primitive-shaped objects, namely, generalized cuboids and generalized cylinders.

[1]outputs instance mask of object body(shape of the whole body) and profile(top view and bottom view faces) using Geonet which fuses Mask-RCNN (state-of-art method for instance segmentation) and deformable convolutional network(DCN). The outputs are labelled as cuboid profile, cuboid body, cylinder profile, and cylinder body. Profiles and bodies are used to predict 3D-profile concerning camera pose. Parallelly, the trajectory axis (a planar 3D curve) which is the main axis (straight or curve) of the body is classified using a neural network with pruning. This axis will be used for profile sweeping along the body to create the 3D model. [1] used 11657 images in the dataset which were further split into 8183 training set and 3474 testing set. Basic architecture of the system is shown in Fig -2.1.

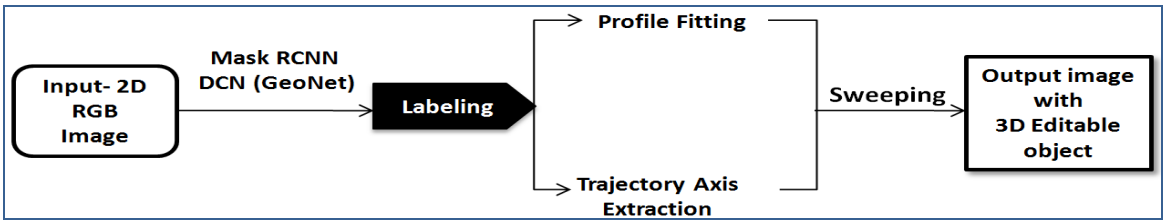

Fig -2.1: System Architecture [1]

According to [1], it took 1 second for GeoNet to segment one image and less than 1 second to reconstruct objects from the masks including stages of instance labeling, profile fitting, and 3D sweeping under a multi-threaded environment. Compared to other existing techniques for 3D reconstructions which construct either point clouds, voxels, or meshes, [1] is able to recover high-quality semantic parts and their relations in the object.

2.2 Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction [2]

[2] have tried to implement an efficient framework for the reconstruction of the 3D object from single RGB image input. According to [2] the prior approaches used voxels based 3D reconstruction which can be easily done using convolutional neural networks(CNN) but results in wasteful representation concerning the fine details, amount of computational power, and time required. On other hand, point clouds are a feasible representation but cannot be directly used with CNNs. Thus, the approach aims to generate dense point clouds in 3D space. The system predicts the output 3D object shapes with dense point clouds from multiple viewpoints. The architecture consists of three phases- 2D structure generator, point cloud fusion, and pseudo rendering. The structure generator consists of convolution layers following linear layers. Two types of convolution operation are used- encoder convolution(halving feature maps) and generator convolution(doubling feature maps). The structure generator outputs eight depth 2D projected images with respect to fixed different viewpoints within a cube. These images are then fused in the point cloud fusion phase to generate a dense point cloud. This is an important and easy step as the viewpoints are static and predefined.

Pseudo renderer renders new depth images at novel viewpoints by using the output point cloud model from the fusion phase. The rendering process is differential which helps to calculate the loss and helps to back-propagate using gradients. This trains the model to make use of loss and generate better point clouds. The geometric reasoning ability of the pseudo renderer lessens the number of trainable/learnable parameters, decreasing the training cost. Average point-wise 3D Euclidean distance between the predicted model and CAD model has been used as a quality metric. The system is trained and evaluated with the ShapeNet database consisting of 3D CAD models. For each 3D model, they have pre-rendered 100 depth and mask images at random viewpoints which will be used to evaluate the loss while the input images are pre-rendered at fixed depth and viewpoints at certain angles. The system architecture of the mentioned algorithm is shown in Fig -2.2.

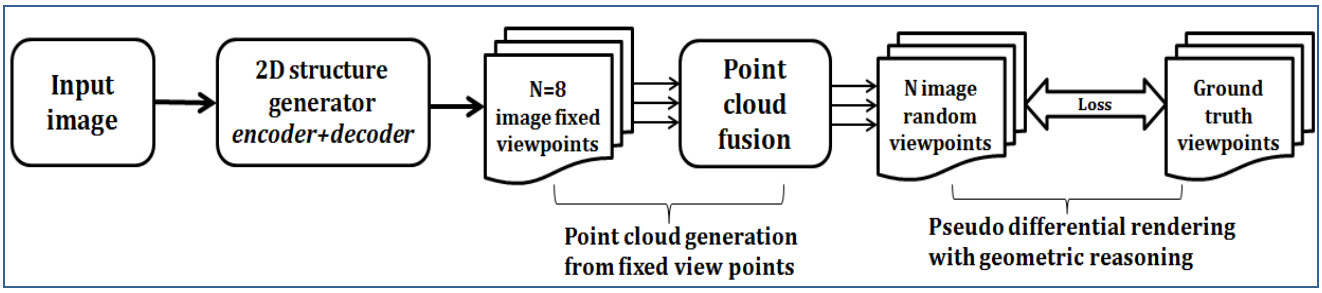

Fig -2.2: System Architecture [2]

2.3 Pixel2Mesh [3]

[3] approach extracts 3D triangular meshes from a single RGB image. The technique uses a graph convolution network(GCN) which represents the 3D Mesh object as a collection of vertices, edges, and faces. G = (V, E, F). [3] provides an end-to-end framework consisting of a network that is divided as an image feature network and a cascaded mesh deformation network. Initially, the system starts with an ellipsoid 3D mesh. The goal is to deform this ellipsoid according to the image features. These image features are extracted by an image feature network(a 2D convolution network) which will deform the initial ellipsoid model into the desired 3D shape. The cascaded mesh deformation network(a GCN) inputs the current deformed mesh from the image feature network and produces new vertices and other mesh features. This deformation network uses graph unpooling layers to increase the density(number) of vertices which will leverage the system to handle fine details with respect to mesh topology.

The system tries to move from coarse level to fine level 3D mesh. [3] have considered three-loss techniques related to meshes viz surface normal loss, laplacian regularization loss, and edge length loss along with champer loss. [3] uses Chamfer Distance (CD) and Earth Mover's Distance (EMD) for evaluation of the model. The basic architectural overview of the system is presented in Fig -2.3. The system receives input images of size 224 × 224, and an initial ellipsoid with 156 vertices and 462 edges. The system takes 15.58ms to reconstruct the mesh with 2466 vertices from a single RGB image. This method fails to construct genus0 objects and holes in the object which could be a major drawback. Also, this method can produce the meshes with the same topology as selected for the initial mesh.

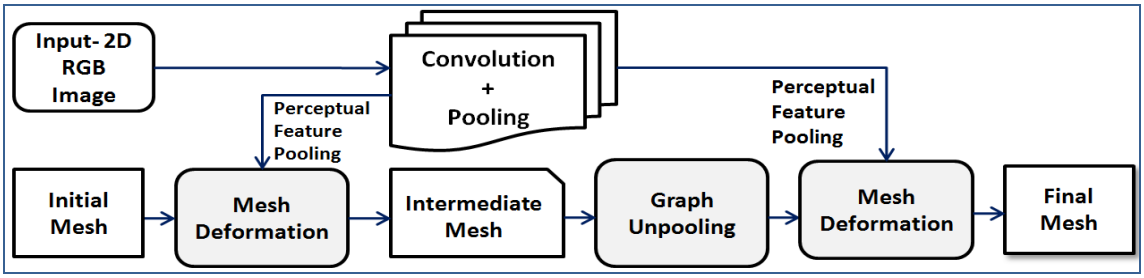

Fig -2.3: System Architecture [3]

2.4 Mesh RCNN [4]

The authors of the paper, 'Mesh RCNN', proposed a technique that extends Mask R-CNN with a mesh prediction branch which gives output meshes having arbitrary topological structure. It first predicts coarse voxel representations that are supposed to be converted to meshes and then refined with vertical alignment and a graph convolution network operating over the mesh vertices and edges.

The method mentioned in [4] is presented in Fig -2.4. It begins with accepting a single input RGB image. The mask branch shown in Fig -2.4 performs 2D recognition using Mask RCNN where 2D objects are detected with bounding boxes, and segmentation masks are generated. The voxel branch first predicts a 3D coarse voxelization of an object, where it produces a 3D grid of occupancy probabilities giving the coarse shape of the object, which is then converted to an initial triangle mesh [4]. This coarse voxel representation has a varying topological structure as it tries to predict mesh with arbitrary topology. This overcomes the problem of fixed or limited object topologies that were present in [3]. Convert the voxelized representation to the mesh and refine the mesh using the iterative mesh refinement technique mentioned in [3].

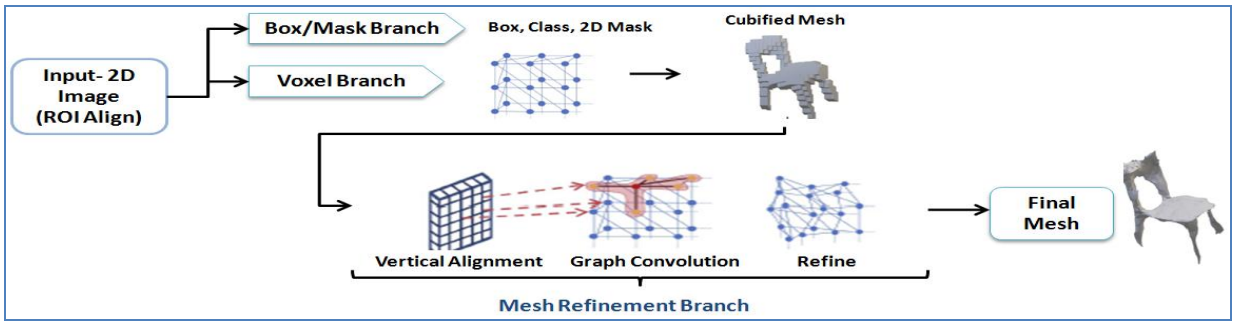

Fig -2.4: System Architecture [4]

[4] further says that the cubify method is applied to the output given by the voxel branch. The cubify inputs voxel occupancy probabilities and a threshold for binarizing voxel occupancy. Each occupied voxel is replaced with a cuboid triangular mesh having 8 vertices, 18 edges, and 12 faces. This results in the initial watertight cubified mesh which only provides a coarse 3D shape of the object. The mesh refinement branch processes this initial cubified mesh whose topology depends on the voxel predictions. Mesh refinement branch refines its vertex positions with a sequence of refinement stages. Each refinement stage consists of three basic operations as vertex alignment, graph convolution, and vertex refinement. Vertex alignment extracts image features for vertices; graph convolution propagates information along mesh edges, and vertex refinement updates the vertex positions. Also, this mesh refinement branch is trained to minimize the mean losses across all refinement stages. Hence, the final refined output mesh is generated. The problem with this technique is it requires 3D supervision that is quite expensive to obtain, and it may not be possible in many cases.

2.5 Accelerating 3D Deep Learning with PyTorch3D [5]

Though [4] overcomes the limitation of fixed topology mentioned in [3], it has its limitation which is 3D supervision that faces a huge trade-off between quality and resources. [5] manages to overcome this trade-off by implementing a similar system under 2D supervision. According to [5] there could be different numerous 3D predictions for the same single 2D image. This is because the machine fails to understand the side view of the object from an image from the single view image. [5] approaches to use a 2-view training setup to predict the 3D shape. For every predicted 3D object there are two corresponding silhouette images where the first image is a silhouette of the actual view from the input RGB image and the second is a silhouette with the rotated view. These silhouettes are used as 2D supervision.

Initially, the 3D shape will be predicted from the single 2D RGB image. The rendered silhouette of this 3D shape will be compared to the first ground truth silhouette image. Also, this 3D shape will be rotated by (R,t) and its rendered silhouette will be compared with the second ground truth silhouette. While comparing both the silhouettes, losses are computed. Pytorch3D have developed their own differential renderer which allows them to back-propagate through the network by calculating the gradients concerning losses, predicted shape, camera coordinates, and silhouette. The further differential renderer can be used for incorporating texture details. The training system architecture is shown in Fig -2.5.1.

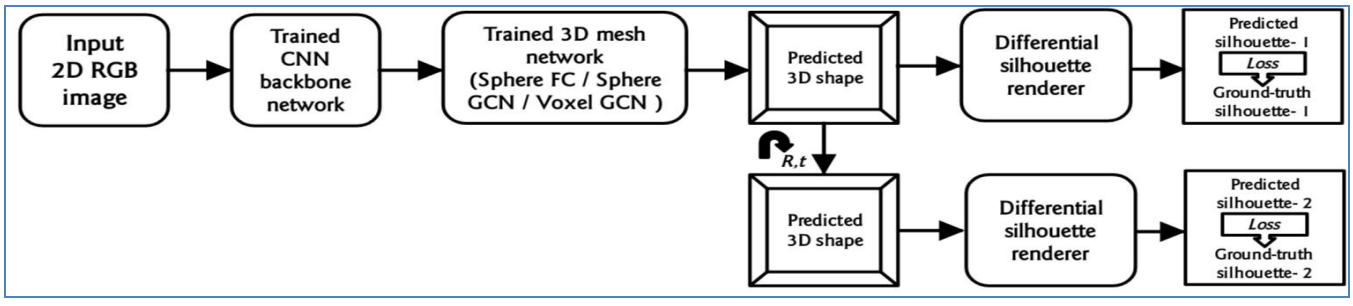

Fig -2.5.1: Training System Architecture [5]

For the prediction of 3D shape, [5] have used Sphere FC, Sphere GCN, and Voxel GCN individually and compared the outputs[3][4]. Sphere FC and Sphere GCN can only make predictions homeomorphic to spheres which is the limitation mentioned in [3]. Voxel GCN can render any genus by topology refinement. The texture is well imposed with Voxel GCN because of regularity in shape. For shading, flat, Phong and Gouraud are compared where flat gives the worst results. [5] have experimented with meshes as well as point clouds. Point Align model method for point clouds is similar to that of mesh which is

mentioned above. The Point Align model is similar to Sphere GCN. With the increase in resolution, the output is well polished. All the above techniques are evaluated based on Champer loss.

With the use of the Pytorch3D library, developed by the authors of [5] there is remarkable speedup in training achieving parallelism with CUDA. The inference system architecture is as shown in Fig -2.5.2.

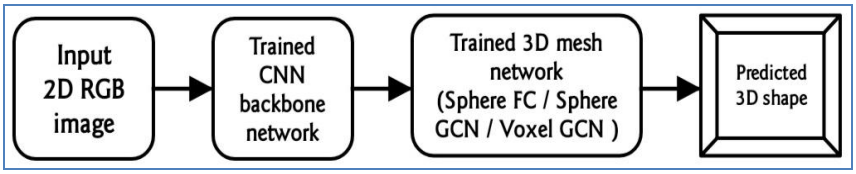

Fig -2.5.2: Inference System Architecture[5]

The comparative study and their respective limitations of all surveyed techniques is mentioned in the Table -1.

3. COMPARATIVE OVERVIEW

Table -1: Comparative study of surveyed papers

Sr. No.	Paper Title	Data structure	Technique/ algorithm	Dataset	Supervision	Limitations
[1]	Autosweep: Recovering 3d editable objects from a single photograph.	Not mentioned	Mask -RCNN, DCN, LeNet, 3-Sweep[6]	ShapeNet, synthesized data, SUN primitive dataset, self-created	2D supervision	Unable to handle objects with spiral axis, regions of instances under occlusion
[2]	Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction	Point Cloud	Image encoder(convolution, linear layers), Structure generator(linear, deconvolution layers), geometric reasoning, Pseudo differential renderer.	Pre-rendered 2D depth/mask image pairs of 3D CAD model from ShapeNet	2D supervision using 2D projections of 3D CAD models	Requires multiple pre-rendered 2D depth/mask images of 3D models in training which can lead to erroneous training if not rendered correctly.
[3]	Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images	Mesh	2D- CNN, Graph based ResNet(G-Resnet), Graph convolution(GCN), vertical alignment	ShapeNet 3D models	3D supervision	Fails to construct genus0 objects(holes), requires fixed topology.
[4]	Mesh RCNN	Mesh, Voxel	Graph convolution(GCN), vertical alignment, ROI alignment, Mesh rcnn (augmented Mask-RCNN)	Pix3D, ShapeNet.	3D supervision	Requires 3D supervision which is expensive,cubify(voxelization) may be computational intensive with increase in classes.
[5]	Accelerating 3D Deep Learning with PyTorch3D	Mesh, Point Cloud	Sphere GCN, Voxel GCN, Sphere FC, silhouette rendering, KNN.	ShapeNetCoreV1 rendered 2D images	2D supervision	Requires 2-view images while training.

4. CONCLUSION

Our paper presented a comparative and comprehensive study of various techniques of 'Automated 2D Image to 3D Model Construction' that has discovered a wide range of available techniques and popular algorithms differing in their generalization, accuracy, complexity, and usability. The comparative study is given in Table -1. Even if the 2D image to 3D Model conversion is a new field, it is evolving rapidly, by the continuous development of new tools meant to simplify the techniques. Recently, the Pytorch3D library brought in differential renderers for meshes and point clouds, data structures to handle meshes and point clouds, and various other functionalities that will speed up the process and make the development of 3D reconstruction techniques easier. This will allow researchers to develop more novel techniques and propose new algorithms that can feasibly convert 2D images to 3D models. Through the comprehensive survey of all the algorithms and techniques given by the mentioned papers, it is possible to build an accurate model that outputs the 3D model with decent accuracy in shape, color, and texture details.

REFERENCES

[1] Xin, Chen, et al. "Autosweep: Recovering 3d editable objects from a single photograph." IEEE transactions on visualization and computer graphics (2018).

[2] Lin, Chen-Hsuan, Chen Kong, and Simon Lucey. "Learning efficient point cloud generation for dense 3d object reconstruction." arXiv preprint arXiv:1706.07036 (2017).

[3] Wang, Nanyang, et al. "Pixel2mesh: Generating 3d mesh models from single rgb images." Proceedings of the European Conference on Computer Vision (ECCV). 2018.

[4] Gkioxari, Georgia, Jitendra Malik, and Justin Johnson. "Mesh r-cnn." Proceedings of the IEEE International Conference on Computer Vision. 2019.

[5] Ravi, Nikhila, et al. "Accelerating 3d deep learning with pytorch3d." arXiv preprint arXiv:2007.08501 (2020).

[6] Chen, Tao, et al. "3-sweep: Extracting editable objects from a single photo." ACM Transactions on Graphics (TOG) 32.6 (2013): 1-10.

[7] Gezawa, Abubakar Sulaiman, et al. "A Review on Deep Learning Approaches for 3D Data Representations in Retrieval and Classifications." IEEE Access 8 (2020): 57566-57593.

[8] Kato, Hiroharu, et al. "Differentiable rendering: A survey." arXiv preprint arXiv:2006.12057 (2020).