Otto-von-Guericke University Magdeburg

Faculty of Computer Science

Master's Thesis

Synth2Real : 3D-Furniture Reconstruction in Ersatz Environment (S2R:3D-FREE)

Author:

Kartik Prabhu

October 15, 2021

Advisors:

Prof. X

National Aeronautics and Space Administration

Dr. No

Department of Technical and Business Information Systems

Otto-von-Guericke University Magdeburg

Prabhu, Kartik:
Synth2Real : 3D-Furniture Reconstruction in Ersatz Environment (S2R:3D-FREE)
Master's Thesis, Otto-von-Guericke University Magdeburg, 2021.

Abstract

Besides the title, the abstract is the most important part of your thesis, as most readers will only read title and abstract. Your goal is to advertise the rest of the thesis for potential readers. For that, you briefly explain what you are focusing on in the thesis. With a misleading abstract, you will miss interested readers and maybe even attract readers with wrong assumptions about your work which will stop reading soon. The abstract should describe the general area as well as the most interesting insights of the work. It is crucial to find the right level of abstraction and length. An abstract typically consists of one paragraph that is significantly shorter than the introduction.

Abstracts typically follow the same structure. You start by describing the research area as well as the general and the specific problem you are focusing on. Then, you outline how you approach the problem in terms of concepts and evaluations. Finally, you close with the most interesting insights that you gained and why they are relevant for the research area.

Acknowledgments

If you want to, you can thank your advisors and/or anyone else who supported you during the thesis in some way.

Contents

List of Figures

List of Tables

List of Code Listings

List of Acronyms

FOL First-Order Logic
SLR Systematic Literature Review
SPL Software Product Line

List of Symbols

General

$\alpha[x\backslash y]$	Substitution of x in α with y
$\mathcal{P}(A)$	Power set of a set A
$f\vert_{A'}$	Restriction of a function $f\colon A \to B$ to a smaller domain $A' \subseteq A$

First-Order and Dynamic Logic

$A \wedge B$	Formalization of the sentence "P and Q"

1. Introduction

As Deep Learning field goes deeper and wider, the need for abundant data has become the basic requirement. The performance of computer vision based on deep learning models heavily depends on the consistency of labelled images. But in practice, collecting data manually is time-consuming, expensive and needs lot of effort. As a solution, we can train models on synthetic dataset and deploy the models to real world problems. Unfortunately it is seen that with supervised learning, the model does not generalise when the distributions of dataset diverges and hence a model trained on synthetic dataset may not perform consistently with a real-world data.

Synthetic data can be defined as data created artificially using computer simulations or algorithms and not from real world events. These data can be created in abundance and hence are frontrunners for Deep Learning training. Synthetic approaches are also means of getting data which might be difficult to collect and annotate in real world. Further more, companies have realised that synthetic data are a necessity not just for task which are inconvinient to annotate, but also for improving the performance of existing models. In a research article published by Gartner **?**, they claim that by 2030, synthetic datasets will overshadow real dataset in all domains. The progression of synthetic data generation is predicted to be as shown in 1.1

Deep Learning has made leaps and bounds in solving 2D-image tasks like classification, segmentations, object detection, etc. It has now entered the realm of Three-Dimension. An image is a 3D objects projects onto a 2D-surface. While doing so, some data from higher dimension is being lost in lower dimension. This inverse ability to reconstruct 3D objects from 2D images has a wide scale application in computer vision and robotics. Tasks like human body shapes and face reconstructions **?????**, 3D-Scene reconstructions **????**, generic object mesh reconstruction with textures are some of the advancements we had over recent years. Here comes the challenge;3D data is not easy to collect. 3D data is not just limited but also noisy and expensive to collect. The results of models depend on the consistency of the labelled data. In this thesis we discuss how game engines can contribute to generating quality labeled dataset for 3D reconstruction task.

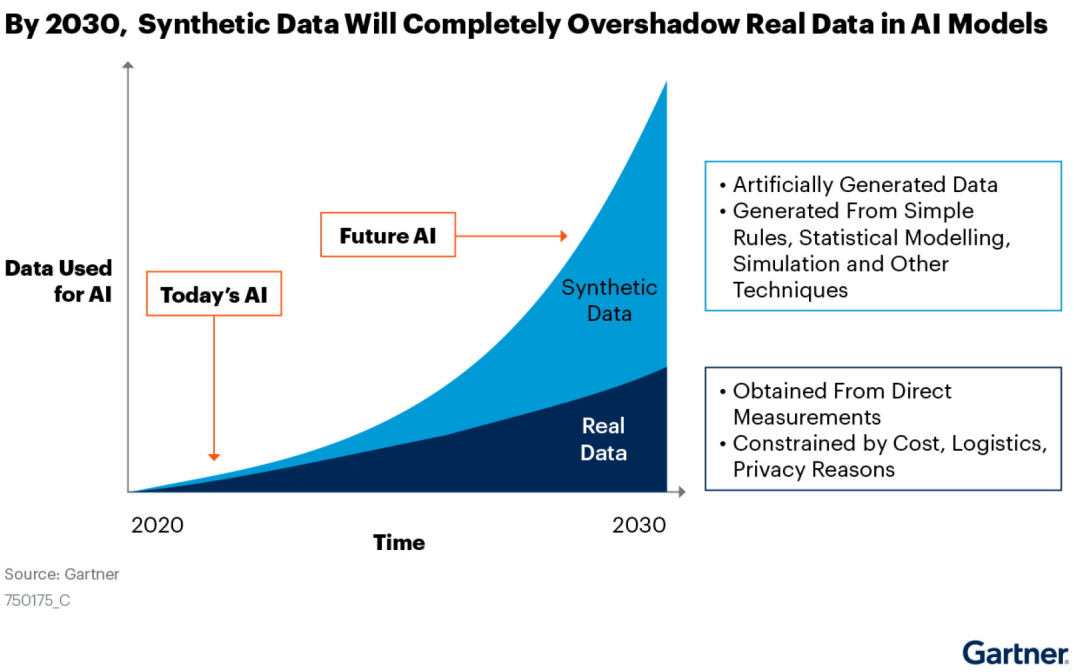


Figure 1.1: A graph showing how synthetic dataset will evolve overtime **?**

1.1 Domain Adaptation vs Domain randomisation

As mentioned earlier, the success of Convolutional Neural Networks depends highly on the training data. As it so happens, not every task will have abundant data to train with. One solution for this problem is using a pre-trained network of similar problem and using transfer learning. The pre-trained network is then trained over with the limited data available for the problem at hand. In this case the features learnt from larger dataset is fine-tuned by overwriting few layers using small annotated dataset of a different domain. Domain adaptation is field of study in machine learning where the distribution of training data differs from testing or target distribution. One shallow approach is as mentioned above; re-weighting the network with testing data to adapt to the problem domain **?**. *add more citations for domain adaptation*. "Deep domain adaptation methods leverage deep networks to learn more transferable representations by embedding domain adaptation in the pipeline of deep learning" **?** For this thesis, domain adaptation is relevant as synthetic dataset might not represent the real or target domain, which can lead to drop in performance.

Domain randomisation on the other hand, is improving data quality such that the target domain distribution is included in the training domain. This is mainly used to bridge the 'reality gap' as termed in **?**, between simulated environment and real images. This is achecived by randomising the rendering of objects in the synthetic data. The key concept here is that by rendering objects with large variance, the real-world distribution will appear to be just a variation in the dataset. Some of the features that can be randomised in a synthetic dataset are textures, brightness, shadows, camera settings, object pose, etc. This will be further discussed in chapter *mention chapter*.

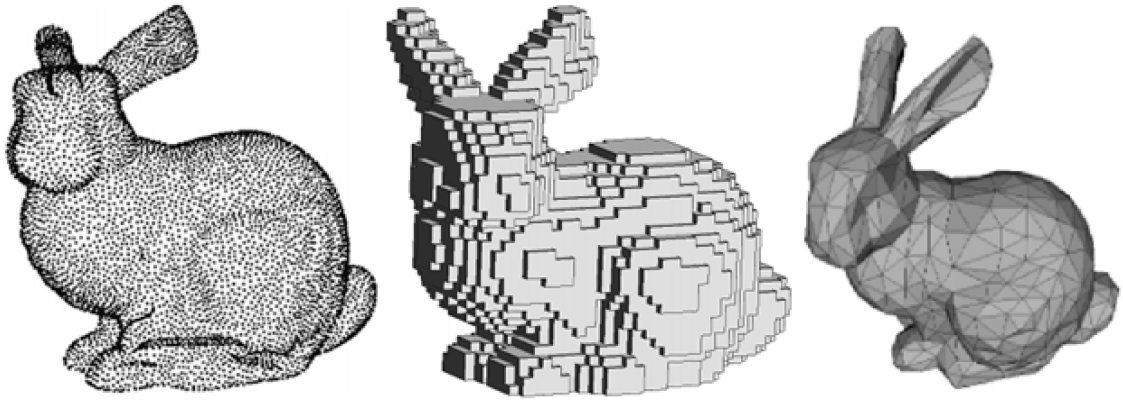


Figure 1.2: 3D representation of Standford bunny model.(left to right) Point cloud, voel and mesh **?**

1.2 Volumetric representations of 3D shapes

The universal representation of images in computer graphics is via pixels. But unlike 2D, 3D data can be represented in various formats like voxels, mesh and point clouds. A sample of thes 3 formats is shown in 1.2. Each of the representation have its own sets of advantages and disadvantages. Voxels, which is short for Volumetric Pixels is a 3D-grid of pixels of constant size. As indicated in **?**, the main advantage of voxels is that Convolution Neural Networks can be easily applied to 3D as in 2D. But since most of the 3D geometry is surface based, it can be wasteful and computationally expensive. Mesh is composed of vertices, edges and faces in 3D space that indicate the formation of 3D objects. This form of representation is far more compact at granular level depending on the resolution. Point clouds on the other hand are collection of 3D points with (x,y,z) coordinates on the surface of the object. The collection of points determines detailing the 3D object representation. But in both these cases CNN is not directly applicable. In this thesis we will be dealing with voxel based models as the focus is on performance of synthetic data rather than the model or the 3d representation itself. *add an image with 3 types of represntations.*

1.3 GameEngines

As per **?**, game engine is defined as "A series of modules and interfaces that allows a development team to focus on product gameplay content, rather than technical content". In layman term game engines are software used for development of games. Unreal Engine, Unity, GameMaker, Amazon Lumeryard, CryEngine are some of the popular game engines in modern times. Over time, the graphics in each of the game engines has improved to an extent where visually we are not able to differentiate between reality and graphics. These use object-oriented techniques which help in modularity. Also, they are GUI-oriented and hence are user friendly and accessible even to a novice programmer. The key component in game engines which we will be focusing on is the 'Rendering engine'. 3D Rendering is process of converting 3D data into 2D images. Rendering can either be real-time or offline/pre-rendering. Game engines aim to make photorealism at highest level with least possible rendering time.

1.4 Background and motivation

3D reconstruction is of great significance to understand scene and to cross the 2D realm. But the task is not trivial, considering the input is still a Two-dimensional RGB image. It is even more challenging since obtaining new data is expensive and time-consuming. The 3D reconstruction field has datasets like ShapeNet **?**, 3D-Front **?**. But there are no corresponding real-images to check if the model will work in real world scenarios. Also searching for real-world examples or creating a set-up scene and then labelling the images manually can lead to human mistakes resulting in wrong labels. A solution to this problem is automating the task of generating synthetic data which can produce pixel-perfect annotations. This can solve human-in-the-loop problem, and while time gets saved, large quantity of data can be generated. But quantity should not the only focus while creating the data, equal importance needs to be given to quality of the data. Game engines have come long way from just supporting 2D to supporting games with photorealistic rendering. Even though game engines have grown to be successful, not many tools are based on it. As 3D models available, we can import these models into the game engine and create an ersatz environment to generate photorealistic dataset. Further, we can check if these photorealisitc synthetic data can be used for 3D reconstruction task.

1.5 Goal of this Thesis

The following research questions will be answered:

1. Are game engines a good medium to create photorealistic sythetic dataset?
   (a) *Experiment: Conduct a survey on real vs synthetic. Use dataset like Pix3d(real), blender proc, unity, 3d-front (synthetic)*
   (b) *Depending on the survey, the most real dataset will be considered as baseline and other datasets will be compared to it.*
2. Can Ersatz environment from a game engine like Unity replace real data for training in 3d reconstruction task?
   (a) *Train on s2r:3dfree data, test on pix3d*
3. To what extent can the performance of model Pre-trained with real dataset improved with Synthetic dataset from game engine, with size of synthetic dataset being 10*(depends on how much data is generated)* times that of real dataset?
   (a) *pretrained on pix3d, continue training on s2r:3dfree*
4. *(optional: Can the photorealism of synthetic image further be enhanced using domain adaptation/GAN techniques?)*

Overall dataset experiments:

1. *Ablation study on chairs, conduct all above mentioned experiments only on chairs as it has highest number of 3d models available*

2. *Finetune chair dataset and check if dataset can be improved to get better results*
3. *Scoremaps on voxel output to see if the probabilty of voxel is high in required region and to see which region is least to give possible explaination fo low IOU*

1.6 Structure of this Thesis

The rest of this thesis is structured as follows:

- Chapter 2, we visit some of the related work on synthetic data generation and state of the art 3d reconstruction networks.
- Chapter 3 deals with concepts and design choices. x
- In Chapter 4, we discuss in detail the 3DScene tool developed using Unity game engine for creating ersatz environment.
- Chapter 5 is dedicated to reviewing and discussing evaluation results obtained by comparison of synthetic and real dataset.
- Finally in Chapter 6, we conclude this thesis with results of the study, highlight few limitations and discuss future improvements.

2. Related Work

2.1 State of the art for 3d-reconstruction

Voxel based, mesh based Pix2vox,OctNet,Mesh rcnn,Pixel2Mesh,Occupancy networks, etc

2.2 Synthetic dataset generation

Synthetic datasets available Explain briefly Scenenet,Openrooms,Hyperism,3dfront,NVIDIA Deep learning Dataset Synthesizer (NDDS),Habitat: A Platform for Embodied AI Research,BlenderProc: Reducing the Reality Gap with Photorealistic Rendering,SynthDet: An end-to-end object detection pipeline using synthetic data None of them seem to provide 3d reconstruction except for 3dfront for some extent. A table to explain diffs between various methods(only if 3D-FREE has more advantage)

2.3 Domain adaptation

Discuss other methods used to mitigate domain shift Example: distance learning, subspace matching

2.3.1 GAN based style transfer

3. Concept

In this section we discuss about dataset and design choice,and the rationale behind reducing the domain gap.

3.1 Pix3D: A large-scale benchmark

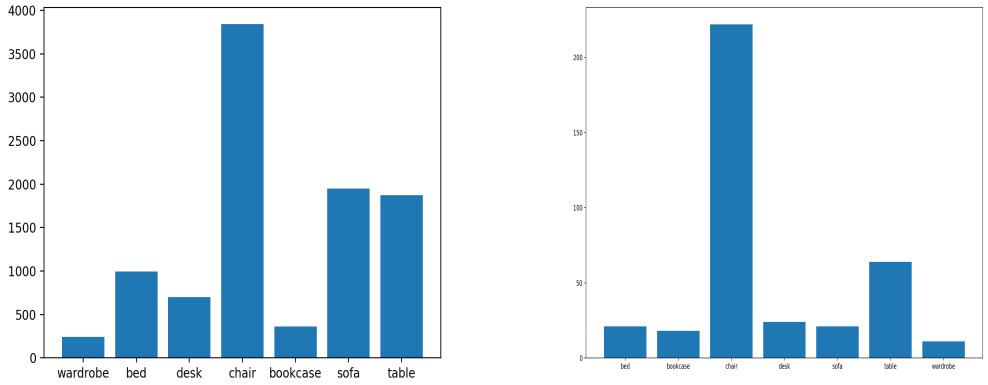
In **?**, a large-scale benchmark for 2D-3D alignment was introduced. The raw images from web search engines were collected and labelled keypoints were used to align the 2D images with the corresponding 3D shapes. The 3D models are extension from IKEA dataset **?**, which is a collection of high-quality IKEA furniture. The dataset also provides with masks and keypoints for the object under observation. For adding more images to the IKEA dataset, the authors of **?** conducted manual web search on Google, Bing, and Baidu, using Ikea model name as keywords. to get around 104,220 images which were further filtered by removing irrelevant images with the help of Amazon Mechanical Turk (AMT) workers. After this manual experiment the total images in Pix3D, only 14,600 images were selected for the 219 Ikea models. For our experiment on synthetic to real dataset, we chose to select only furniture classes from Pix3D, leaving out "misc and tools" classes, which were significantly less to begin with.

3.1.1 Disadvantages of Pix3D

The distribution of models in pix3d is as shown in 3.1. As we can see, the dataset distribution is uneven across classes and more than 50% of classes have less than 1000 images.

Though Pix3D set a benchmark for 2D-3D alignment, here are few disadvantages of using this real dataset.

1. For Deep Learning approches, we need large-scale data and 14,600 might not be sufficient.
2. The orientation of object is not randomised.
3. The dataset does not provide 2.5D information (i.e. depth and normal) which can be crucial for 3D learning.



(a) **(b)**

Figure 3.1: Distribution of pix3D **?** images(left), unique models(right)

3.1.2 Why Pix3D?

Pix3D is a collection of indoor scenes with complex background, varying light conditions with shadows, reflective surfaces and even varying occlusion levels .Each image comprises of collection of objects in the scene, but only one object from the catagory is annotated. It is a perfect example of having limited real world data. And since 3D models are available for each furniture,a synthetic data can be generated in abundance from those models.

3.2 Role of SceneNet

SceneNet **?** is a large collection photorealistic indoor scene trajectories. The dataset provides with images and videos of indoor scene which can be used for tasks like SLAM, semantic and instance segmentation, object detection and further enhanced for other vision problems like optical flow depth and pose estimation **?**). They use ShapeNet **?** models to occupy 57 indoor scenes which gives the scene unlimited configurations. Unfortunately there is no mapping from scene to 3D model for tasks like 3D reconstructions. In our approach, we utilize the scene provided by SceneNet as layout for our indoor scenes. This includes initial shapenet models with furniture placement. We then replace the class under observation with a corresponding model from Pix3D.

3.3 Unity based pipeline

Unity game engine is used for creating the synthetic dataset. Unity is a cross-platform game engine, the basic version of which is available for free. There is no official comparison between Unreal Engine and Unity engine to have a deciding factor. The selection of unity engine was purely individual choice and ease of use. But both these game engines have ability to create realistic looking scenes which can be used for synthetic dataset generation. With Unity game engine, the available scenes from scenenet and 3D models from pix3d can be imported to form an ersatz environment.

(a) **(b)**

(c) **(d)**

Figure 3.2: Sample RGB images from pix3D

Further domain randomisation can be applied to create a dataset of photorealistic images and 2.5D data like normal, depthmaps, masks.

3.3.1 Domain Randomisation with Unity Engine

In this section we discuss what kind of domain randomisation is applied for the dataset creation.

3.3.1.1 Scenes from scenenet

As discussed in section 3.2, SceneNet provides wide variety of indoor scenes which includes bedrooms, bathroom, living room, kitchen and office. For our case, we did not consider bathroom as the class under observations are furnitures and we rarely have any furnitures in the bathroom. For the rest of the scenes we have a pre-determined setup given by scenenet. We use 25 scenes in total as basic rooms for our target object to be place. And with further randomisation we feel this should be sufficient to check if unity can produce a useful dataset.

3.3.1.2 Camera Viewpoints

The position and orientation of camera gives us the most randomisation in this setup. The minimum and maximum distance can be chosen by the user for the camera to be places from the target object. A random point can be selected in this range as the position for the camera. As for the orientation of the camera, we can make the camera to always look at the target object, irrespiective of the position. With

this, we acheived different background for target object, and the target object itself appears to be in different position and orientation.

3.3.1.3 Lighting and shadows

Lighting plays an important role in photorealism. If the lighting is not setup correctly the synthetic dataset will never seem to be photorealistic. Unity offers wide variety of lighting like global light which acts like the sunlight, and various indoor lighting systems. Ideally we should make the luminous objects like lamps, chandlier, bulbs etc be the source of light for indoor scenes, but we observed that the room does not lit up uniformly which makes in less photorealistic. Hence we use some pre-determined lighting settings which will be further discussed in implementation section. *mention section*.

3.3.1.4 Randomised textures

SceneNet **?** also provides with textures for different categories in the scene. We then further increase the texture database by adding few more textures from ambientCG.com, which provide license for free. We randomly allocate textures to each object making sure that each category has same texture to make the scene more uniform.

3.3.1.5 Replacing target objects

To further randomise the scene, the category of target objects in the scene are replaced by the object under observation. When there are more than one object of same category we randomise the selection of which object is to be replaced which further randomises the captured data. The scale of the target object is made to match the scale of the category object in the scene in the least dimension. For example if the length of the length of the category object in the scene is the least amonth length,width and height, then the target object is scaled to match this length. This makes the target object blend in with the scene.

3.4 S2R:3D-FREE, a Pix3D based sythetic dataset

S2R:3D-FREE dataset, which stands for Synth2Real: 3-Dimensional Furniture Reconstruction from Ersatz Environment, is a combination of SceneNet and Pix3D dataset. We utilize the avalaibilty of 3D models of rooms and furntitures from these 2 dataset to create an ersatz environment using Unity as our framework. We randomise the indoor scenes from SceneNet **?** along with textures provided by them with some additional complex textures. The other option would have been to use the shapenet as the target model, but in this case we would not be having a real dataset to compare to. Ideally a model trained on shapenet should also perform well with a real dataset like pix3d, but we decided to have a synthetic dataset based on pix3d itseld for better comparison.

3.5 3D reconstruction pipeline

Now that we discussed on how the dataset will be created, we move to the deep learning aspect of this thesis.

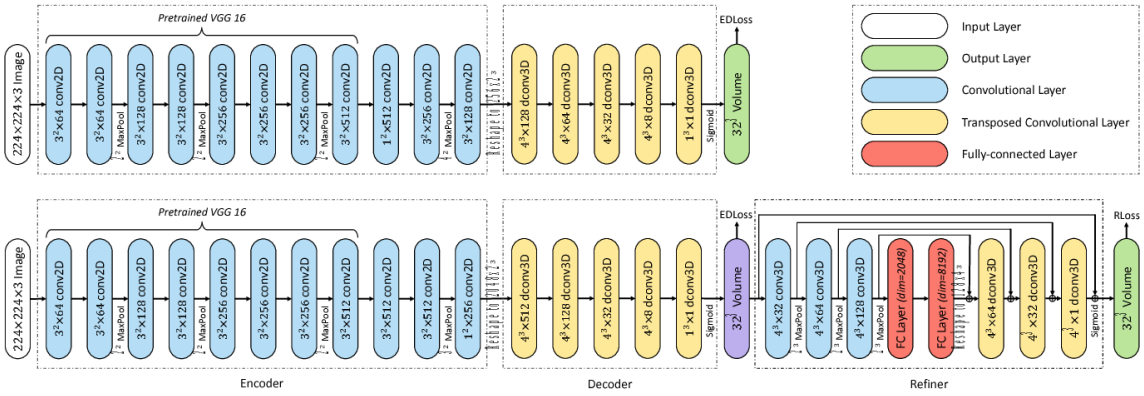


Figure 3.3: Network architecture for pix2vox **?**

3.5.1 Pix2Vox

The architecture of pix2vox **?** as in 3.3. The network is comprised of 4 modules: Encoder, Decoder, Merger and Refiner. Merger plays significant part when it comes to mult-view reconstruction of 3d objects. But we focus on single-view 3d reconstruction and hence merger will not influence the output significantly. As Encoder, pix2vox has utilised VGG16 network **?** pre-trained on ImageNet **?**. Decoder is an expansive network which converts 2D embeddings to 3D voxel grid. Refiner is an auto-encoder which takes the 3D output from the merger and produces more refined final output.

3.5.1.1 Why Pix2Vox?

Pix2Vox has been used as baseline by most of the research oriented to 3D reconstruction. This network is one of the few networks to be tested on pix3d dataset. According to the survey conducted by **?**, performance of pix2vox **?** has be noted to be significantly higher when compared to previous work *cite other papers* as shown in 3.4. While this comparison was made on 3d reconstuction of shapenet dataset, since pix3d was not available at the time when previous work were published. From our survey, only CoReNet **?** had a slight gain in performance compared to pix2vox. When trained on shapenet and tested with pix3d, CoReNet gave a result of 29.7% IoU while Pix2Vox gave a result of 28.8% IoU and Pix2Vox++ a result of 29.2% IoU. Since the difference in the performance was not statistically significant we decided to stick with the baseline model itself. Another reason for selecting pix2vox model is that the backbone of the architecture is pre-trained with ImageNet and hence the embeddings generated from this encoder can help in visualising the domain space of both Pix3D (real images) and S2R:3DFREE(synthetic images). *write about pix2vox++, which uses resnet as backbone if used*. Furthermore, the focus of this thesis is not to check which is the best model to reconstruct the model, but to check if game engines can produce photorealistic images usable for 3d reconstruction. Hence the selection of the model was not of utmost importance.

3.6 Domain adaptation with Gan *if we decide to go with this approach*

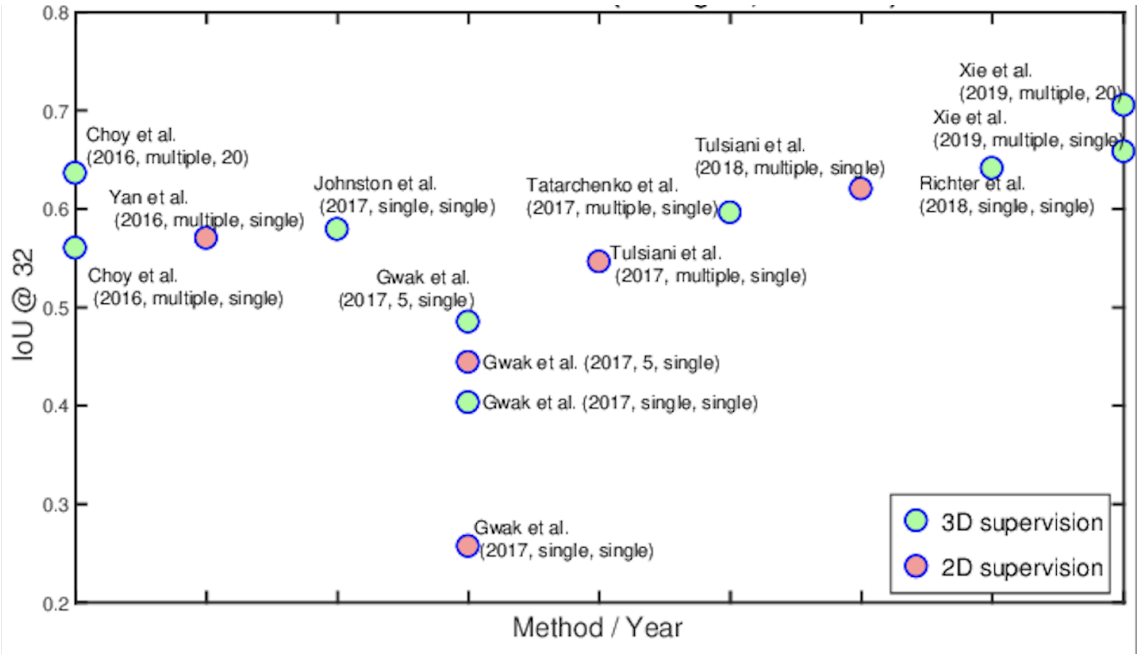


Figure 3.4: A survey conducted by **?**, proves that Pix2Vox is considerably a good 3D reconstruction model. *replace this figure with better one*

4. Implementation

4.1 3D-Scene framework

Replacing models, lighting(indoor and outdoor), randomisation of texture, camera distance, camera angles, outdoor skybox, UI controls Manual inputs, automated Unity - ML-ImageSynthesis UML diagram is needed? (If it helps)

4.2 3d-Reconstruction framework

Development environment Training setup(hardware, voxel size...) Mixed precision using apex If mixed precision helped performance, memory, speed, evaluation metric

4.3 Implement a domain adaptation technique

To serve as a proof if more domain gap reduction is needed

5. Evaluation

5.1 Domain gaps - Qualitative

Visualize domain gap between S2R:3D-FREE and real image(pix3d) Visualize domain gap between other synthetic dataset like 3dfront, scenenet tsne

5.2 Domain gaps - Quantitative

Maximum Mean Discrepancy Kullback-Leibler divergence

5.3 Performance

Performance of model trained on synthetic dataset by testing model with real dataset(pix3d) Qualitative: Voxel output comparisons Quantitative: IOU, Dice

5.4 Domain shift technique

Compare the differences in domain shift of models learnt on S2R:3D-FREE and a traditional domain shift learning.

5.5 Ablation study on chairs

Check if randomising textures of chair and background helps improve performance. Example: Create dataset with constant room texture vs randomising texture, Check if lighting helps (indoor, outdoor). Example: Create a dataset with constant lighting(only outdoor lighting) vs indoor lighting. Iou per category: as in https://arxiv.org/pdf/1905.03678.pdf

6. Conclusion

6.1 Summary

6.2 Limitations of game engines

6.3 Future Work

Appendix

This table is automatically generated from a CSV file. Of course, you can also create tables from scratch.

	List_ Insert	List_ Search	Ordered_ Insert	Ordered_ Search	Ordered_ Min	Ordered_ Sort	Set_ Insert
coarse, Ex., Strict	2400	1418	10608	10824	2871	10169	23366
coarse, Ex., Def.	2400	1418	9652	20172	2871	14043	35884
coarse, Ex., None	2400	1418	14347	20160	2871	12687	54907
coarse, L.S., Strict	2400	1418	10608	10824	2871	10169	21047
coarse, L.S., Default	2400	1418	9430	20172	2871	13835	24688
coarse, L.S., None	2400	1418	13802	20160	2871	12687	28501
coarse, Complete	2400	1418	4504	10149	2871	10047	16114
coarse, Product	9600	2836	9010	20298	5742	20094	16114
fine, Ex., Strict	2211	1321	11210	16984	2501	6500	12901
fine, Ex., Def.	2211	1321	10299	16971	2501	6420	14695
fine, Ex., None	2211	1321	10881	16836	2501	7115	23005
fine, L.S., Strict	2211	1321	11210	16984	2501	6500	12301
fine, L.S., Default	2211	1321	10251	16971	2501	6420	13173
fine, L.S., None	2211	1321	11116	16836	2501	7115	20053
fine, Complete	2211	1321	2018	16531	2501	6613	9336
fine, Product	8844	2642	4036	33062	5002	13226	9336

Figure A.1: Caption in the text.

B. Tutorial

Acronyms like software product lines (SPLs) and first-order logic (FOL) are automatically added to the list of acronyms above. To reference a chapter, subsection, figure, etc., put a label after the command (see above) and then use Chapter 1.

You can mark stuff as TODO, which is useful for writing drafts.[1]

Or put notes to your advisors in the margin.

Theorem B.1: My Cool Theorem

You can also add definitions and theorems in mathematical theses.

Proof. You can also add proofs, though most theses do not require any. □

In Figure B.5, we show an example of how to typeset a complex figure only with LaTeX by using TikZ.

B.1 This is a section

For citing, put entries in the BibTeX file (one example is there) and use the following to reference the authors directly: **?** conducted a systematic literature review (SLR) (this is an acronym defined in the main file), or use the following for references in parenthesis [**?**]. **?** **?**

B.1.1 This is a subsection

Numerated list

1. One
2. Two
3. Three

[1] Footnotes are also possible, but are rarely used in computer science.

0.29

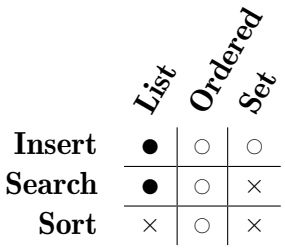


Figure B.1: List

0.21

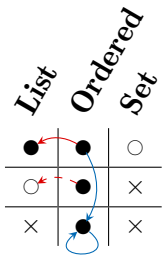


Figure B.2: List, Ordered

0.19

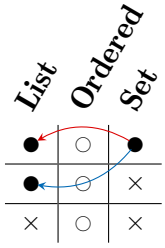


Figure B.3: List, Set

0.26

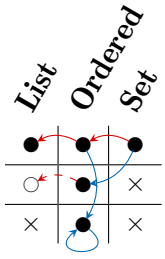


Figure B.4: List, Ordered, Set

Figure B.5: Drawing with tables and TikZ.

Table B.1: Long caption for a table.

Column1	Column2	Column3	Column4
Left	Right	Centered	Left, fixed width
		Multi column	
Multi row	X Y		

Bullet list

- One
- Two
- Three

I herewith assure that I wrote the present thesis independently, that the thesis has not been partially or fully submitted as graded academic work and that I have used no other means than the ones indicated. I have indicated all parts of the work in which sources are used according to their wording or to their meaning.

Magdeburg, 15th October 2021