University of Glasgow | School of Computing Science

Honours Individual Project Dissertation

Exploring 3D reconstruction from RGB images of partially occluded objects

Davide Greco

August 18, 2023

Abstract

3D reconstruction from partially occluded images has been rarely investigated. This project aims to explore the impact of occlusion on Pix2Vox, a multi-view reconstruction model, and possible approaches for improving its generated 3D shapes. In this work erasing data augmentation techniques have been used for generating occluded versions of the ShapeNet dataset and StableDiffusion for inpainting the occluded images before feeding them to Pix2Vox. Results show that the developed approach improves the performance of the baseline by 51% on severe occlusion. Interestingly, fine-tuning the Pix2Vox model using occluded images by 30% to 40%, further improves the performance by 97% on severe occlusion. Additionally, the approach improves the generalisation and performance on unseen real-world images by 16%.

Acknowledgements

I would like to thank my supervisors Dr Gethin Norman and Dr Edmond S. L Ho for supporting and guiding me through this project.

Education Use Consent

I hereby grant my permission for this project to be stored, distributed and shown to other University of Glasgow students and staff for educational purposes. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Signature: Davide Greco Date: 18 Aug 2023

Contents

1 Introduction

3D reconstruction is the process of generating a 3D shape given a series of input images. It has many applications, ranging from the medical field (e.g., surgical planning and prosthetics (Rengier et al. 2010)), to the entertainment industry (e.g., visual fitting and special effects production (Fu et al. 2021)). Single-view 3D reconstruction tries to achieve the same results by leveraging information from only a single image, often from uncalibrated cameras and in various light conditions. The advantage of this approach is the flexibility of applying 3D reconstruction in cases where the environment cannot be controlled and multiple images are hard or impossible to retrieve. This project explores the case in which the object is affected by inter-class occlusion, a type of partial occlusion caused by another entity. Since the current research does not investigate this case extensively, there is the need for generating a new occluded dataset. Hence, we propose a set of occluded variations of the ShapeNet (Chang et al. 2015) dataset, generated by erasing part of the original images. The images in the dataset differentiate depending on the schemes used for applying erasing, the erasing area applied, and the location of the erasing mask. Moreover, we proposed approaches to improve Pix2Vox 3D reconstruction of under partial occlusion, which include fine-tuning the 3D reconstruction model and inpainting the input images. Results show that fine-tuning the pre-trained Pix2Vox model on occluded images achieves the highest results, performing 97% better than the baseline in case of significant occlusion. On the other hand, applying inpainting increases the performance of the pre-trained Pix2Vox model by 51% in scenarios involving severe occlusion.

Dissertation structure. The structure of the dissertation is divided into the following seven chapters.

- **Chapter 2** explores the field of 3D reconstruction to identify the current state-of-the-art and the research gap. In the field of image data augmentation methods for identifying the methods for generating an occluded dataset are discussed. Finally, it the field of inpainting is explored, with a focus on state-of-the-art deep learning methods.
- **Chapter 3** discusses the challenges of 3D reconstruction such as partial occlusion and the lack of occluded data.
- **Chapter 4** outlines the design of the proposed approaches for generating new occluded datasets and for improving the 3D reconstruction process using inpainting and fine-tuning.
- **Chapter 5** explains the erasing method used for generating the occluded datasets. Following, it explains the inpainting and 3D reconstruction models and the chosen hyper-parameters.
- **Chapter 6** analyses the quality of the generated datasets in terms of information loss. Next, the baseline 3D reconstruction model is compared to the fine-tuned models on the proposed approaches. Additionally, the limitations of the proposed methods and future work are discussed.
- **Chapter 7** summarises the project and highlights the most important obtained results.

2 Background

While the scope of this project, which is to improve the results of Pix2Vox under occlusion, is narrow and specific, the fields of 3D reconstruction, data augmentation and inpainting are extensive. Hence in Section 2.1, 3D reconstruction approaches will be narrowed down to only those that rely on RGB or 2.5D images. The discussions and issues included will focus on deep learning approaches that aim to reconstruct a single 3D shape and volumetric representations. Section 2.2 covers data augmentation and it will discuss the different techniques that can be applied to the case of occlusion. Finally, in Section 2.3, only the various deep learning approaches for inpainting will be explored, which represent the current state-of-the-art in this area.

2.1 3D reconstruction

Three-dimensional (3D) reconstruction is an ill-posed problem which aims to deduce a 3D model given a single image, multiple images or a video (Han et al. 2019; Fu et al. 2021). In the case of a video, this can be seen as a sequence of images with a spatial-temporal correlation (Han et al. 2019).

Formally, in the case of multiple images, given a set $I = \{I_1, I_2, \ldots, I_n\}$ of RGB images and the ground truth shape X, the goal of the 3D reconstruction process is to minimise the loss L represented as the difference between the predicted shape X^* and the unknown, true shape X (Han et al. 2019; Fu et al. 2021).

Since the goal is the reconstruction of a 3D shape, every side or view needs to be reconstructed, even those that cannot be seen from a single or multiple 2D images. Hence there may be multiple solutions and for this reason, it is an ill-posed inverse problem. To help the reconstruction process, additional information can be included such as silhouettes, segmentation masks, and semantic labels (Han et al. 2019).

Applications. Since 3D reconstruction has many applications in various fields (e.g., robotics and medicine), the aim goes beyond reconstructing a single object (Han et al. 2019; Fu et al. 2021). Indeed, it could be used to reconstruct an entire 3D scene for robot navigation and scene understanding. In the case of medical diagnosis, the purpose could be to reconstruct anatomic parts, faces or entire bodies. Additionally, there are multiple other applications, e.g., for 3D modelling and animation (Han et al. 2019; Fu et al. 2021).

For the remainder of this chapter, the term "subject" will refer to the object, scene, human body, part or face that wants to be reconstructed. Since the different applications will use different ways of representing the relative 3D shape, we will use the term "3D shape" or "3D geometry" to reference the shape generally.

Traditional methods. Traditional methods exploit the mathematics of the 3D to 2D projection process, to solve the inverse problem. They are based on prior assumptions or 2D annotations and are often limited to a single class of subject (Han et al. 2019; Fu et al. 2021). For example, Dovgard and Basri (2004) proposed a single least-squares system of equations to reconstruct a human face from a single image. However, the illumination has to be known and from the side.

Due to these restricting assumptions, these traditional methods are difficult to apply to real-world situations. On the other hand, they do not rely on huge datasets as deep learning methods do. Most traditional methods need multiple images from well-calibrated cameras to reach effective results. Additionally, the collection of images has to depict as much of the object as possible without leaving huge areas uncovered (Lowe 2004). Han et al. (2019) categorise traditional methods into two types: *stereo-based* and *shape-from-silhouette*. Given multiple images at different angles, stereo-based approaches match the common features and then reconstruct the 3D shape using triangulation. On the other hand, shape-from-silhouette approaches use multiple segmented 2D silhouettes from well-calibrated cameras to generate accurate 3D shapes.

Deep Learning methods. The current generation of approaches for 3D reconstruction leverage prior knowledge, and hence formulate the 3D reconstruction problem as a recognition problem (Han et al. 2019). Similar to what humans do when they leverage their experience and prior knowledge in approximating the 3D structure and guessing the hidden parts of it (Han et al. 2019). These methods are better suited than traditional methods for real-world applications since require only a single image or just a few images (Tulsiani et al. 2017) that are not necessarily taken from well-calibrated cameras (Han et al. 2019).

The main network architecture used in deep learning methods is autoencoder (AE). All autoencoders have two components: an encoder, which maps an input to latent space, and a decoder, which maps latent space to an output. The latent space is a compressed and meaningful representation of the input. In this case, the encoder-decoder architecture is used to generate a 3D geometry from one or more images. Hence, the encoder is a 2D encoder, since takes as input 2D images, extracts the key features from the images and encodes them into a latent space. The decoder is a 3D decoder, which from the latent space generates a 3D geometry. Thus, this network architecture is considered a 3D autoencoder (3D-AE).

The 3D autoencoder network architecture is the backbone of this research area, on which different and substantial variations have been created. The 2D encoders are grouped depending on the generated latent space, while the 3D decoders will be categorised based on the type of representation of the 3D geometry they generate, for example, voxels, point clouds and meshes (Han et al. 2019; Fu et al. 2021).

2.1.1 2D Encoders

2D Encoders can be grouped depending on the latent space they encode the features into and therefore are divided into *discrete*, *continuous* and *intermediate* encodings (Fu et al. 2021).

Discrete encodings. An autoencoder has a discrete latent space if its encoder maps the input into a feature vector of fixed length (Choy et al. 2016; Girdhar et al. 2016; Han et al. 2019; Fu et al. 2021). For example, Girdhar et al. (2016) encodes RGB input images into a 64-D vector embedding. The drawbacks are that it does not allow for an easy interpolation, and it does not guarantee that a small perturbation of the input implies a small perturbation in the output (Han et al. 2019). Figure 2.1 shows a diagram of a discrete 2D encoder (Fu et al. 2021).

Intermediate encodings. Intermediate encodings are a subcategory of discrete encodings. As well as discrete encodings, intermediate encodings encode images into a fixed feature vector; the difference is that in intermediate encoding, there are two encoders (Wu et al. 2017; Zhang et al. 2018; Fu et al. 2021). The first encoder maps the input into an intermediate representation. Then, the second encoder maps the intermediate representation into the discrete latent space (Fu et al. 2021). Examples are the work of Wu et al. (2017) and Zhang et al. (2018), which are discussed in Section 2.1.3. Figure 2.2 shows diagrams of two intermediate 2D encoders (Fu et al. 2021).

Continuous encodings. Autoencoders have a continuous latent space if its encoder maps the input into a mean vector (μ) and a vector of standard deviation (σ) of a multivariate Gaussian distribution (Kingma and Welling 2013; Wu et al. 2016; Smith and Meger 2017; Han et al. 2019;

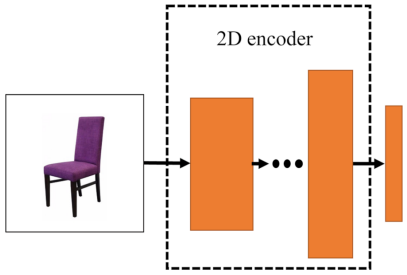


Figure 2.1: *In discrete encoding, a 2D encoder directly encodes the input image into a latent vector of fixed length (Fu et al. 2021).*

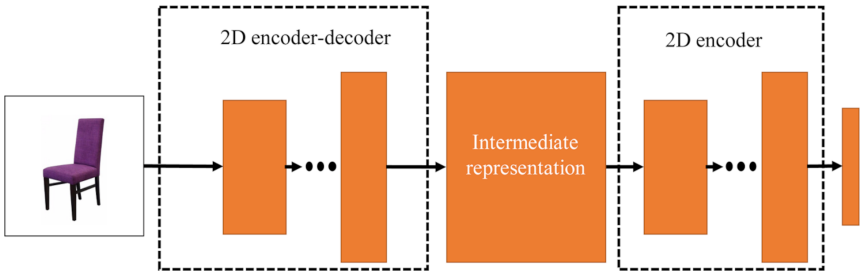


Figure 2.2: *In intermediate encoding, the first 2D encoder encodes the input image into an intermediate representation, then the second encoder encodes the intermediate representation into discrete latent space (Fu et al. 2021).*

Fu et al. 2021). These types of autoencoders are defined as Variational Autoencoders (VAE) or 3D-VAE in the case of a 3D decoder. Using a distribution instead of a fixed length vector as a latent space allows for easy sampling and interpolation (Han et al. 2019). Furthermore, 3D-VAE can randomly sample from the latent space to generate multiple variations of the 3D reconstructed shape. 3D reconstruction is an ill-posed problem that does not have a single solution. Hence, having a model that allows one to choose from multiple plausible reconstructions is an advantage compared to previous methods that allowed a single, fixed solution. Additionally, this also means generalising better to newly seen images (Han et al. 2019; Fu et al. 2021). Figure 2.3 shows a diagram of a continuous 2D encoder (Fu et al. 2021).

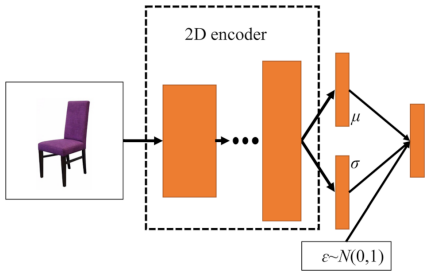


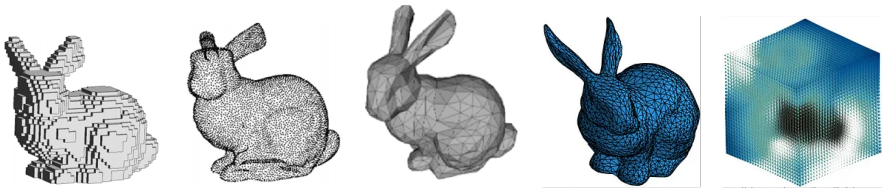
Figure 2.3: *In continuous encoding, a 2D encoder directly encodes the input image into a continuous latent space (Fu et al. 2021).*

2.1.2 3D Decoders

As Han et al. (2019) highlight, the choice of the type of representation of the final 3D geometry, and hence of the 3D decoder, is fundamental. Indeed, this has consequences on the type of

network architecture to use, the quality of the result, the computational efficiency and on the possible application of the final model. Needless to say that each representation has its benefits and drawbacks. Following, there will be a brief explanation of *high-level* and *low-level* 3D decoders. Then, the rest of the section will explore the different types of low-level 3D encoders and their differences. In particular, low-level 3D encoders can represent their output in discrete space using *voxels*, *point clouds* and *meshes*, and continuous space using *parametric* and *implicit* methods (Fu et al. 2021).

When talking about high-level and low-level 3D representations, Fu et al. (2021) refer to the level of features encoded in the latent space from which the decoder will generate the 3D geometry. High-level features regard the structure of the single subjects and the arrangement of the scene. The output of such decoders is composed of volumetric or surface primitives, which better retain the structure of the subject but are less detailed. Low-level features regard, on the other hand, the details of the subjects such as contours, edges, angles, and colours. Low-level 3D decoders estimate a more detailed output, however, they lack a general understanding of the subject's structure. Low-level 3D decoders can be categorised depending on how they represent the produced 3D geometries, which could be using voxel grids, point clouds or meshes for discrete representations, and parametric and implicit methods for continuous representations (see Figure 2.4).



(a) Voxels *(b) Point clouds* *(c) Mesh* *(d) Parametric (as triangular mesh)* *(e) Implicit*

Figure 2.4: *Types of low-level 3D representations: (a) voxels (Hoang et al. 2019), (b) point clouds (Hoang et al. 2019), (c) mesh (Hoang et al. 2019), (d) parametric representation as a tringulated mesh (Michalkiewicz et al. 2019), and (e) implicit representation (Michalkiewicz et al. 2019).*

Voxel grid representations. A voxel grid is a 3D grid that subdivides space into cubes (or voxels), transforming it from a continuous to a discrete domain. Hence, voxels can be seen as pixels in a 3D space. Each voxel will contain some data, for example, the probability of it being part of the object. Voxel 3D reconstruction aims to generate a 3D voxel grid that is similar to the ground truth. Voxel representation has the advantage of being easily implemented. Indeed, a 2D convolutional neural network (CNN) can be easily converted to a 3D CNN that supports volumetric representation. Additionally, it can reconstruct subjects with any arbitrary topological structure (Fu et al. 2021).

However, 3D decoders using voxel representation are expensive in terms of memory requirements; indeed, common voxel grid sizes, also referred as voxel resolutions, are between 32^3 and 128^3 (Girdhar et al. 2016; Choy et al. 2016; Wu et al. 2017; Tulsiani et al. 2017; Xie et al. 2019) and, for particularly efficient approaches, 256^3 and 512^3 (Häne et al. 2017; Tatarchenko et al. 2017). Nonetheless, a voxel grid of size 512^3 is often not enough to achieve fine-grained resolution (Han et al. 2019; Fu et al. 2021). Section 2.1.3 will further discuss this type of representation.

Point cloud representations. Similarly to voxel representations, point cloud representations are simple and highly flexible (Fu et al. 2021). While their representation is memory-efficient, they use fully connected layers, hence are computationally expensive. Indeed, point clouds have an irregular structure and cannot be easily used with 3D CNN. (Han et al. 2019; Fu et al. 2021). However, fully connected layers have the advantage of better capturing global information Han

et al. (2019).

Mesh-based representations. Mesh-based representations use vertices and faces to represent surfaces. Mesh-based representations are similar to point cloud representations in being memory-efficient (Han et al. 2019) and not being compatible with traditional 3D CNN (Han et al. 2019; Fu et al. 2021).

Implicit representations. Implicit representations classify each 3D point as being outside, inside or on the surface S (Yadav 2022; Fu et al. 2021; Han et al. 2019). The classification function F returns positive values for points outside the object, negative values for points inside the object and zero for points on the surface S(Yadav 2022; Fu et al. 2021). It follows that the surface S can be defined by the zero outputs of the function F (Yadav 2022; Fu et al. 2021). While this method generates 3D geometries with superior visual quality, they have a long training and inference time (Fu et al. 2021).

Parametric representations. Similarly to implicit representations, parametric representations represent the 3D surface S as a function F that maps the parameter domain P to the surface S (Sinha et al. 2017; Groueix et al. 2018; Yadav 2022). Parametric representation leverage this mapping to reduce the 3D reconstruction problem to a 2D problem of the parameter domain (Sinha et al. 2017; Groueix et al. 2018; Yadav 2022; Han et al. 2019). 3D surfaces are then generated using the resulting parametric representation(Sinha et al. 2017; Groueix et al. 2018).

2.1.3 Voxel grid representations

The 3D decoding phase of voxel representations can be categorised depending on the additional processes or changes applied to it. Figure 2.5 shows the diagrams of dense, intermediate and octree voxel decodings (Fu et al. 2021).

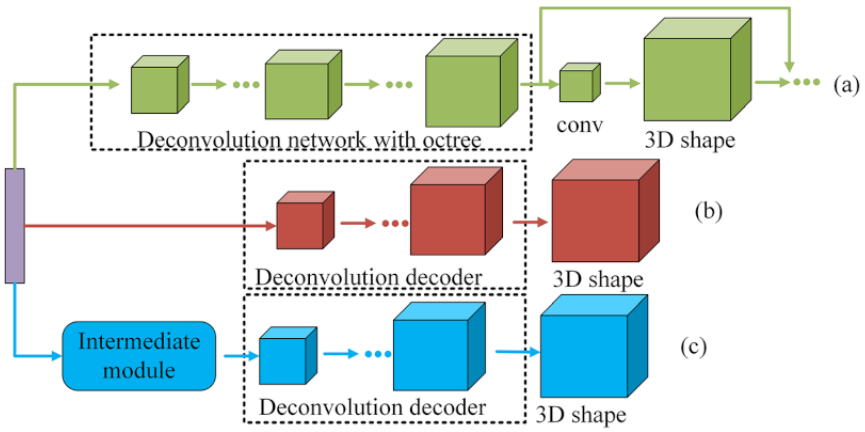


Figure 2.5: *The diagrams of (a) octree voxel decoding, (b) dense voxel decoding and (c) intermediate voxel decoding (Fu et al. 2021).*

Dense voxel decoding. This category represents the methods that from the latent space infer the voxel grid through the direct use of 3D convolutions (Wu et al. 2014; Girdhar et al. 2016; Tulsiani et al. 2017). Wu et al. (2014) was the first to build a 3D deep learning model. Their method uses a *Convolutional Deep Belief Network* and 2.5D depth maps to both recognise and reconstruct an object. Their network learns the joint distribution of all 3D voxels and represents the geometric 3D shape as a probability distribution of binary variables on a 3D voxel grid. As a first work, their resolution is limited to a 30^2 voxel grid.

Girdhar et al. (2016) propose a *TL-embedding Network* composed of two components: a 3D autoencoder to strengthen the 3D generative ability of the network and a CNN to reinforce the

representativeness of the latent space. During training, the 3D autoencoder learns to correctly reconstruct 3D voxels, while the CNN learns to map the given 2D images into encodings. This is the T-network. During the test phase, the 3D encoder is removed and the CNN is connected to the 3D decoder. This is the L-network. While this work performs well on the IKEA Ben-Shabat et al. (2020) and PASCAL 3D+ Xiang et al. (2014) datasets (Girdhar et al. 2016), the voxel grid resolution is only 20^3, even lower than Wu et al. (2014), the first work in this regard.

Pix2Vox, proposed by Xie et al. (2019), supports 3D reconstruction from single or multiple images and uses a context-aware fusion module and a refiner to achieve state-of-the-art 3D voxel representation. The idea of first estimating a low-resolution or coarse voxel grid and subsequently refining it is not unique to this work and will be used in other occasions as discussed further below. In this case, Pix2Vox (Xie et al. 2019) tries to exploit as much as possible the additional information provided by multiple images to create a first coarse volume and then refine it. In particular, for each image is inferred a relative coarse representation. The context-aware fusion module assigns a score to every voxel of each representation, depending on the quality of the reconstruction. Hence, the module fuses them into one representation that will be then fed to the refiner. The refiner, which can be seen as a residual network, will then produce the final volume.

Intermediate voxel decoding. The key insight behind intermediate voxel decodings is to extract additional information from the input images that the 3D decoders can use effectively. For example, Wu et al. (2017) proposes *MarrNet* which, given an input RGB image, estimates the relative 2.5D sketches (depth, normal maps and silhouette) using a 2D autoencoder. Then, these 2.5D sketches are fed to a 3D autoencoder that infers the final 3D shape. The major advantage is that the final trained model suffers less of the domain adaptation problem. The reason is that the 3D autoencoder does not uses the initial RGB images, but only the estimated 2.5D sketches. These sketches can be generated both from real-world and synthetic images and do not include object appearance variations such as lighting. The biggest issue with this approach is that correctly estimating 2.5D sketches is challenging and often results in distortions (Xie et al. 2019).

Zhang et al. (2018) builds on the MarrNet framework and introduces *Generalizable Reconstruction* (GenRe), which uses spherical maps to perform 3D reconstruction. As with MarrNet, a depth image is estimated from the input 2D image, however, it is not directly used for prediction of the 3D shape. Instead, the depth image is projected into a spherical map, which will be partial due to self-occlusion. Hence, a full spherical map is estimated using inpainting. Finally, the 3D voxel is estimated from the inpainted spherical map, and then further refined. As with the case of MarrNet (Wu et al. 2017), using 2.5D sketches and spherical maps helps in the generalisation of the network. In this case, GenRe (Zhang et al. 2018), at the time of its publication, achieved state-of-the-art performance in 3D reconstruction both on seen and unseen classes.

Sparse Octree decoding. One of the biggest disadvantages of voxel grids is the memory requirement to store them. Indeed, a voxel grid of 256^3, which is a resolution above the average of the seen methods, contains about 16.7 million voxels, which Tatarchenko et al. (2017) estimates as 9.98 GB of space. Another approach would be to divide the grid space using sparse octrees. An octree is a tree data structure where each node has exactly 8 children. The idea is to subdivide the space grid into several voxels that are multiple of 8, making it representable by an octree tree structure. Hence, each node's child will represent a voxel and will have a value of 0 if it does not contain any part of the 3D object, or a non-zero positive number if it does. It follows that the detail of the 3D voxel representation can be set by deciding a specified height of the tree. However, this does not address the memory issue of voxel representation, hence the use of a sparse octree structure. In a sparse octree, the majority of its nodes are empty (null), hence it allows nodes without children, making it a memory-optimised representation compared to octrees and previously seen voxel grids.

However, as noted by Han et al. (2019), there are two main issues with this approach, which appear due to the irregular nature of the representation. The first is that convolutional operations

are easier to implement on regular grids. To tackle this issue, Wang (2017) proposes O-CNN: an octree data structure that supports memory and execution on GPU, allowing for efficient storage of the octant and CNN information. The second is that the octree structure depends on the specific object being represented, hence the network will have to estimate both the octree structure and its content.

Häne et al. (2017) and Tatarchenko et al. (2017) proposed similar methods that, starting from low resolutions such as 32^3, progressively estimate a higher resolution of both the octree structure and the voxel grid at the same time.

Häne et al. (2017) presents the Hierarchical Surface Prediction (HSP) which explores the octree using a depth-first algorithm and achieves a resolution of 256^3. Tatarchenko et al. (2017) proposes the Octree Generating Network (OGN) which explores the octree using a breadth-first algorithm and reaches a resolution of 512^3. Additionally, Tatarchenko et al. (2017) shows that while traditional dense approaches scale cubically, their approach scales quadratically, both in memory requirements and runtime. However, in practice, a difference is only seen from resolutions of 128^3 or higher.

Other voxel decoding. Until now we have described various changes and additions to the 3D autoencoder network architecture. However, there are cases in which 3D autoencoders have been combined with other networks such as *Recurrent Neural Networks* (RNNs) (Choy et al. 2016) and *Generative Adversarial Networks* (GANs) (Goodfellow et al. 2014; Wu et al. 2016; Smith and Meger 2017). For example, Choy et al. (2016), introduces *3D Recurrent Reconstruction Neural Network* (3D-R2N2) for multi-view 3D reconstruction. Each image is encoded into low-dimensional feature vectors, which are combined by the proposed 3D-LSTM module. The core idea is to merge the encodings by retaining past information and refining it as new images are available. Then a 3D-DCNN increases the hidden state of the 3D-LSTM module to the target 3D voxel resolution. It should be noted that the previously explained Pix2Vox (Xie et al. 2019) has a similar aim of Choy et al. (2016). Indeed, both try to have a coarse first representation that will then be updated as new images are added.

Wu et al. (2016) is the first to implement the use of GANs in the field of 3D reconstruction. The idea, first developed by Larsen et al. (2015) using 2D images, is to use the decoder of a VAE as the generator of GAN. Hence, the proposed 3D-VAE-GAN architecture has a 2D encoder to extract a latent representation from the single input image, a 3D decoder (that is also a generator) to reconstruct the 3D shape and a discriminator (D) to classify the produced 3D shape as either real or generated. Even though Smith and Meger (2017) do build on Wu et al. (2016) to try and better stabilize the GAN training, convergence and stability are still significant issues of the GAN method (Fu et al. 2021). More information on the GAN network architecture and its drawbacks is in Section 2.3.2.

2.1.4 Evaluation metrics

In this section, we discuss the most commonly used evaluation metrics for volumetric representations.

Intersection over Unit (IoU) (Xie et al. 2019; Han et al. 2019; Fu et al. 2021). This metric is used to evaluate volumetric and mesh-based representations. However, mesh representation needs to be voxelised first. IoU measures the ratio of the intersection between the predicted voxel grid and the ground truth, to the union of the two voxel grids. The higher the IoU value is, the better the reconstruction. This metric is defined as follows:

$$IoU = \frac{\sum_{i,j,k}[I(x_{(i,j,k)} > t)I(y_{(i,j,k)})]}{\sum_{i,j,k}[I(x_{(i,j,k)} > t) + I(y_{(i,j,k)})]} \tag{2.1}$$

where $x_{(i,j,k)}$ is the predicted occupancy probability and $y_{(i,j,k)}$ is the ground truth at coordinates (i, j, k). Furthermore, $I(\cdot)$ is an indicator function and t is the voxelisation threshold. In other

words, t determines under which probability a voxel has to be considered empty (0) or containing the part of the object (1).

Mean of Cross Entropy Loss (CE) (Han et al. 2019). This metric can be used to evaluate volumetric and point-based representations. This metric is defined as follows:

$$CE = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log x_i + (1-y_i)\log(1-x_i)] \tag{2.2}$$

where N is the total number of voxels or points. x is the estimated value at the ith voxel or point. A lower CE value corresponds to a better 3D reconstruction.

2.1.5 Datasets

Datasets can include a variety of classes, ranging from generic objects to entire scenes or have only specific classes such as cars or birds. Independently on the type of training, any dataset has a collection of 2D images that are either real or synthetically generated, could be of one or more subjects, and with a uniform background or cluttered. Table 2.1 shows a list of the most popular datasets used for 3D voxel reconstruction.

Dataset	Img Type	Classes Type	No. Classes	Background	Subject per img	Img with 3D GT
ShapeNet (Chang et al. 2015)	Synthetic	Generic	55	Uniform	Single	51,300
ModelNet (Wu et al. 2014)	Synthetic	Generic	662	Uniform	Single	127,915
PASCAL3D+ (Xiang et al. 2014)	Real	Generic	12	Cluttered	Multiple	36,000
IKEA (Ben-Shabat et al. 2020)	Real	Furniture	6	Cluttered	Single	219
Pix3D (Sun et al. 2018)	Real	Furniture	9	Cluttered	Single	1015

Table 2.1: *Popular datasets for 3D voxel reconstruction*

2.2 Image Data Augmentation

Data augmentation (DA) approaches generates new data samples from the available ones, to reduce the overfitting of a model and address the problem of limited data (Lewy and Mańdziuk 2022; Shorten and Khoshgoftaar 2019). An overfitted model performs incredibly well on the training data but struggles when faced with unseen data. Hence, overfitting can be seen as the opposite of generalisation, where generalisation can be seen as the ability to use the learnt knowledge during training, to correctly generate an output given new unseen images. One of the core issues of overfitting is a limited, non-representative training dataset (Shorten and Khoshgoftaar 2019). This section will cover data augmentation methods applied to the domain of images, which is pertinent to the scope of the project.

Data augmentation techniques are categorised into *data warping augmentations* and *oversampling augmentations*. Data warping augmentations modify existing images and keep the association with their labels, on the other hand, oversampling augmentations generate new instances to add to the dataset (Lewy and Mańdziuk 2022; Shorten and Khoshgoftaar 2019). Using a method from one category does not exclude the possibility of using a method of the other category it is common to combine multiple methods of both categories.

Safety of data augmentation Shorten and Khoshgoftaar (2019) highlights the important concept of "safety" of the application of data augmentation techniques. The concept refers to whether applying the DA method will preserve or alter its label. For example, rotation is a widely used DA method which is safe if applied to animals: an image of a cat is still an image of a cat even after being rotated by 180 degrees. However, this does not hold for other datasets, such as numeric datasets: an image depicting the number 6, hence paired with the label "six", will depict the number 9 after a rotation of 180 degrees. Hence, in this case, rotation is not a safe data augmentation method.

2.2.1 Basic image manipulation

Basic image manipulation methods are computationally efficient and manipulate the image directly. Basic manipulation methods can be subdivided into image manipulation, image erasing and image mix (Yang et al. 2022a). Emphasis is placed on image erasing methods since are particularly inherent to the project.

Image manipulation. Image manipulation methods are easy to implement, yet effective and versatile Shorten and Khoshgoftaar (2019); Yang et al. (2022a). Additionally, they are considered to be data warping augmentations, since images modified by them will preserve their labels (Shorten and Khoshgoftaar 2019; Yang et al. 2022a). Yang et al. (2022a) conveniently summarised all image manipulation methods and their description in Table 2.2.

Methods	Description
Flipping	Flip the image horizontally, vertically or both.
Rotation	Rotate the image at a specific angle.
Scaling ratio	Increase or reduce the image size.
Noise injection	Add noise into the image.
Colour space	Change the image colour channels.
Contrast	Change the image contrast.
Sharpening	Modify the image sharpness.
Translation	Move horizontally, vertically, or both.
Cropping	Crop a sub-region of the image.

Table 2.2: *List of image manipulation methods and the relative description (Yang et al. 2022a)*

Some methods, such as translation and cropping could seem very similar, however, have a substantial distinction. Translation differs from cropping since it is size-preserving while cropping results in a reduction of image size (Shorten and Khoshgoftaar 2019). Hence, after applying translation, part of the image will be outside the image boundary and lost (Yang et al. 2022a). Since transformation is size-preserving, the lost areas are replaced with a fixed constant, for example, 0 (Yang et al. 2022a). Rotation also suffers from this drawback, which is defined as the padding effect (Yang et al. 2022a).

Image erasing. Image erasing methods are data warping augmentation methods that allow for improving the robustness of a model on partially occluded images (Lewy and Mańdziuk 2022; Zhong et al. 2017), i.e., images where parts of the image that are unclear (Shorten and Khoshgoftaar (2019)). Examples of such methods are Random Erasing (Zhong et al. 2017) and Cutout (DeVries and Taylor 2017a) which erase/mask out a random region of the image by replacing its pixels values with a constant value or random noise. The different effects of Cutout and Random erasing are shown in Figure 2.6 (Lewy and Mańdziuk 2022).

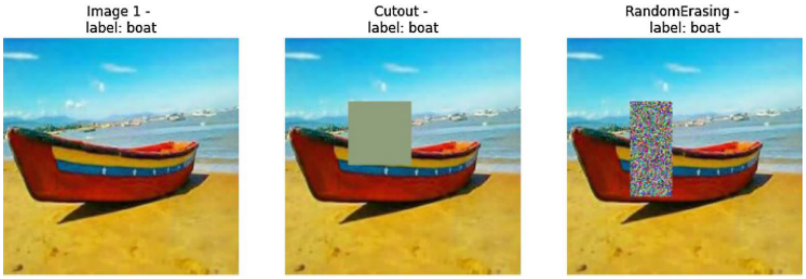


Figure 2.6: *Cutout and Random Erasing applied to an image (Lewy and Mańdziuk 2022)*

Random erasing. Random Erasing randomly occludes a rectangular region of the image by replacing its area with random noise. As with other data augmentation techniques, there is a hyperparameter p, the probability of applying such a method for each image on the dataset during each iteration. Other hyperparameters are the area ratio, which indicates how much of the image will be occluded and the aspect ratio, which indicates the ratio between the height and width of the rectangle region (Zhong et al. 2017).

The rationale is that by occluding a random portion of the image, the model will not overfit a particular feature. Instead, it will be forced to take into consideration the entirety of the image and more descriptive and variegate features as discriminators. Hence, Random Erasing improves the robustness of the CNNs to partially occluded images.

Zhong et al. (2017) propose three different random erasing schemes:

- *Image-Aware Random Erasing* (IRE), which applies Random Erasing on the entire image;
- *Object-Aware Random Erasing* (ORE), which applies Random Erasing inside the bounding boxes of the object;
- *Image and object-aware Random Erasing* (I+ORE), which applies Random Erasing to both the entire image and inside the bounding boxes.

Zhong et al. (2017) reports the results of using these three different schemes in training Fast R-CNN (Girshick 2015). Results show that using ORE is slightly better than using IRE with, respectively, an improvement on the baseline of 1.0% and 0.8%. Using I+ORE outperforms both methods with an improvement over the baseline of 1.4% (Zhong et al. 2017).

Cutout. The Cutout (DeVries and Taylor 2017a) method is similar to Random Erasing (Zhong et al. 2017) since both remove a random continuous region of the image. The differences include that the Cutout region shape is squared, instead of having a varying aspect ratio, fixed-size, instead of having a random area ratio between intervals and, zero masks, instead of filling it with random noise (DeVries and Taylor 2017a).

However, DeVries and Taylor (2017a) add two novelty reflections. The first consideration is that a targeted approach, that masked out the maximally activated features of the feature maps, is not any better than removing random fixed-size regions. The second consideration is that the size of the cutout region is a more important hyperparameter than the shape.

Patch Gaussian. Lopes et al. (2019) combine Cutout with Gaussian noise augmentation, since Cutout improves accuracy but does not ensure robustness, while Gaussian noise increases robustness but at the cost of accuracy. Hence, instead of erasing the Cutout region, is applied Gaussian blur.

Image mix Image mixing techniques consist in combining two or more random training images into a new training sample. Inoue (2018) introduces *SamplePairing*, which, after applying random cropping and horizontal flipping, mixes two images by averaging their pixel values for each RGB channel. SamplePairing is part of the class of linear image mixing methods, which generates a new sample using a linear combination of two images, for example, element-wise averaging. Non-linear or generalised mixing image methods have been explored by Summers and Dinneen (2019), which showed the efficacy of concatenating patches, rows or columns of two sample images into a new training sample (examples are shown in Figure A.1).

Both linear and non-linear image mixing methods are generally non-label preserving since the label of the new mixed image is a combination of the labels of the original images. An exception is SamplePairing (Inoue 2018), which pairs the generated sample with the label of the first of the two original images. Experiments on image mixing methods have been proven to be effective in reducing the error rate when applied (Shorten and Khoshgoftaar 2019; Inoue 2018; Summers and Dinneen 2019). Inoue (2018) showed that their approach is particularly effective when combining images randomly selected from all classes, rather than using images of the same class. The drawback of image mixing is that its efficacy is counter-intuitive from

the human perspective and it is difficult to explain the performance boost obtained by applying it (Shorten and Khoshgoftaar 2019). It makes even more sense after noticing that generalised techniques are effective only if combined with other data augmentations methods (Summers and Dinneen 2019). One possible explanation, suggested by Shorten and Khoshgoftaar (2019), is that by applying image mixing methods, and hence increasing the dataset size, the model will create a more robust representation of low-level features, such as edges.

2.2.2 Deep learning-based image augmentation

Feature space augmentation. Feature space augmentation differs particularly from other image mixing methods since it is applied to the latent space level instead of the input layer. DeVries and Taylor (2017b) method first extracts the low-dimensional representation and applies data augmentation methods such as adding noise, interpolation and extrapolation. Another example is the work of Li et al. (2021), which encourages the use of the moments of latent features during training. Moments are just the mean and standard deviation of latent features, which are often discarded by normalising the latent features. The idea of Li et al. (2021) is to replace the moments of the learned features of one training image with those of another, encouraging their usage in addition to the normalised features.

A disadvantage of feature space augmentation is the difficulty of interpreting original and newly generated vector data (Shorten and Khoshgoftaar 2019). The newly generated features can be converted into images by using an auto-encoder (Shorten and Khoshgoftaar 2019). However, this exponentially increases complexity and training time, especially for deep CNN architectures (Shorten and Khoshgoftaar 2019).

GAN-based data augmentation. Deep learning generative models aim to learn the underlying true data distribution of a collection of images to then generate new data by sampling from learned distribution (Yang et al. 2022a). Needless to say that generative models are extremely useful in the context of image data augmentation. Indeed, they have the potential to address the issue of imbalanced data, which occurs when in a dataset with multiple classes, at least one class has significantly fewer images than the other classes. Lim et al. (2018), for example, use GANs for decreasing the false positive error in unsupervised anomaly detection by oversampling samples that occur with a small probability. GANs (Goodfellow et al. 2014) have been mentioned in the previous section when exploring 3D reconstruction, and will be further explained in Section 2.3, when discussing inpainting approaches. GANs could also address the issue cased when there is a limited initial dataset, as long as it is large enough to train the GAN model (Shorten and Khoshgoftaar 2019). However, if there is a need for image-to-image translation, GANs require a large dataset with paired samples. CycleGANs (Zhu et al. 2017a) tackle this issue by combining two GAN models to perform unpaired image-to-image translation. The idea is that if an image of collection X is translated to Y and then Y is translated back to X, this should result in the same image. Hence they use the first GAN, which given an image of collection X, generates an image of collection Y, then the second GAN will generate an image of collection X, given an image of collection Y. This also holds if an image of collection Y is translated to X and then from X to Y. Zhu et al. (2017b) uses a CycleGAN network to improve the performance of the proposed CNN classifier on emotion recognition. The emotion recognition dataset used, FER2013 (Goodfellow et al. 2013) is an imbalanced collection of seven different emotions. Some classes, such as disgust, have a small number of examples and are referred as minority classes. Zhu et al. (2017b) fix this issue by translating images from other classes into the minority classes.

Neural style transfer. Neural Style Transfer (Gatys et al. 2015) allows disentangling the hidden representation of CNNs to extract the style and content of a given image. The extracted style is then transferred into another image, preserving its content. It is similar to the scope image-to-image problem, however, contrary to CycleGANs (Zhu et al. 2017a), the target is the style of single images, not entire collections Huang et al. (2018). While mostly used in artistic applications,

style transfer is used as a data augmentation method to ease the transition from simulated datasets to the real-world (Shorten and Khoshgoftaar 2019). For example, Tobin et al. (2017) apply different styles to change the texture, lighting and number of objects of the training data for object localisation. They show that adding enough variability in the training data style, causes the object detector to see the real-world case as another variation of the data. However, Shorten and Khoshgoftaar (2019), highlights that creating such a diverse collection of images is expensive in terms of computational and memory resources. On the other hand, Johnson et al. (2016) introduces a faster algorithm, however, it uses a limited pre-trained set of styles. Using a small set of styles, additionally to not providing enough variability, has the risk of adding its biases into the training dataset (Shorten and Khoshgoftaar 2019).

2.3 Inpainting

Inpainting aims to restore missing or unwanted parts of an image in a realistic manner (Jam et al. 2021; Xiang et al. 2023; Parida et al. 2023). Similarly to 3D reconstruction, there are multiple plausible ground truths and hence it is an ill-posed problem. Applications vary from film restoration (Newson et al. 2014), art conservation (Baatz et al. 2008) and artefact removal (Vitoria and Ballester 2019; Tovey et al. 2019).

2.3.1 Traditional methods

The first approaches, defined as traditional methods, tried to exploit surrounding and existent information to fill the missing parts of the image (Qin et al. 2021). Traditional diffusion-based methods worked well for recreating repeated patterns or textures and the removal of small areas of the image due to the small diffusion distance of pixel information around the hole (Qin et al. 2021). However, as the size of the hole increased their effectiveness decreased due to their lack of understanding of the high-level structure and disposition of the image. Hence they did not have success in recreating more complex scenes or in inpainting large sections.

2.3.2 Deep Learning

Modern deep-learning approaches use autoencoders to remedy the lack of a higher semantic understanding of the image. Indeed, the encoder captures the high-level structural and semantic abstractions into latent space, which will be used by the decoder to generate the new image Jam et al. (2021). This is a similar concept that has been illustrated with 3D reconstruction. In this case, since the input and output will both be 2D images, there is no issue with choosing a new way of representing the output as discussed with 3D decoders. The autoencoder network architecture is used in a *Generative Adversarial Network* (GAN) framework which improves the sharpness of the final image (Pathak et al. 2016) and allows for achieving excellent qualitative results (Goodfellow et al. 2014).

Generative adversarial networks (GANs). First proposed by Goodfellow et al. (2014), a GAN network is composed of two modules: a generator (G) and a discriminator (D). When applied to the inpainting task, the generator inpaints the missing region of an input image, generating a fake image. The discriminator is a classifier that assigns a probability to distinguish a generated image from a ground truth image. This improves coherency between the generated images and the original images Jam et al. (2021). Figure 2.7 is an example of a GAN having an encoder-decoder as the generator component and a CNN classifier as the discriminator (Jam et al. 2021).

However, as the training progresses and the generator improves its ability to generate fake images, the discriminator will decrease its ability to correctly discern the two classes. The discriminator will have a vanishing gradient, which will make difficult convergence Jam et al. (2021) and hence train. (Pathak et al. 2016) has been the first to combine AE and GAN for the task of

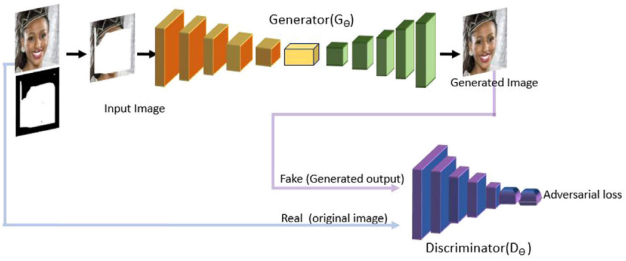


Figure 2.7: *An overview of the encoder-decoder architecture applied in a GAN framework for the task of image inapainting (Jam et al. 2021).*

image inpaining. The approach uses surrounding information to fill in the missing content, a concept similar to traditional methods such as Hays and Efros (2007), but achieving better results. However, while improved, the problem of having structural cohesion in the output remains, since the discriminator focuses on the missing parts of the image and not on the overall context. An additional issue is the lack of edge-preserving results Jam et al. (2021).

Reverse Masking Network (R-MNet), proposed by Jam et al. (2020), tries to overcome, with good results, these issues. Particularly, the issue of blending the newly generated pixels with the visible ones. The idea is to transfer the reversed image at the end of the AE network, making it inpaint only the missing pixels and achieving better-blending results. An additional consequence is that the final inpainted image will keep the original structure and texture.

As pointed out by Dhariwal and Nichol (2021), GANs suffer from various drawbacks, such as the need for specially selected hyperparameters and regularizers to avoid them from collapsing. This makes them harder to scale and apply to new domains and captures less diversity compared to likelihood-based models, we will discuss this further in Section 4.2.2.

Diffusion models. Diffusion models are a type of likelihood-based model that gradually generate images by removing noise from a random signal (Dhariwal and Nichol 2021; Parida et al. 2023). A diffusion model has two processes: the forward process, which iteratively add noise to the image until it becomes all noise, and the reverse process, which iteratively removes noise and estimates the original image (Sohl-Dickstein et al. 2015; Dhariwal and Nichol 2021; Parida et al. 2023).

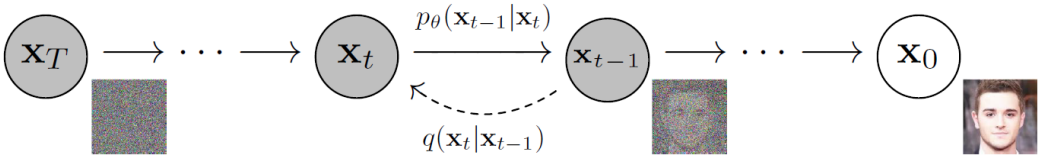


Figure 2.8: *The process of the diffused probabilistic mode introduced by (Ho et al. 2020). $q(x_t|x_{t-1})$ is the forward process, while $p_\theta(x_t|x_{t-1})$ represent the reverse process*

The idea, inspired by thermodynamics, was first introduced by Sohl-Dickstein et al. (2015). Ho et al. (2020) applied the proposed diffusion probabilistic model to high-quality image synthesis. Dhariwal and Nichol (2021) builds on the work of Ho et al. (2020), proposing a *Denoising Diffusion Probabilistic Model* (DDPM) that achieves state-of-the-art results. In Lugmayr et al. (2022) DDPM is used as the refinement step after generating a coarse image based on the missing and surrounding pixels. In addition to further optimising accuracy and speed, the method has the benefit of being flexible, handling various types of damage to the image (such as holes, cracks and missing pixels) and being able to inpaint textured surfaces (such as hair, fur and grass). Nonetheless

the various improvements and optimisations, the main disadvantage of diffusion models lies in the computational requirements and the high training and inference time (Rombach et al. 2021; Dhariwal and Nichol 2021; Parida et al. 2023).

Latent diffusion models. To overcome the computational costs of diffusion models, Rombach et al. (2021) proposed a *latent Diffusion Model* (LDM). The key difference with previous diffusion models is that instead of generating a new image on the pixel space, their method works in a latent space level. Additionally, the LDM can be conditioned by pairing the image with additional inputs, defined as conditional inputs, making it able to fulfil multiple tasks such as image generation, super-resolution and inpainting. In the case of inpainting, the input image can be paired with the masks that define the areas that should be inpainted, and a text prompt that describes what type of content they should be replaced with. The process starts by encoding the input image and the conditional inputs each into the relative low dimensional latent space. Their latent spaces are then merged using an attention-based mechanism. Here there is the biggest change: the merged latent space is the direct input of the diffusion process, without being upsampling first as the other methods. The upsampling is instead the last step, which takes as input the diffused latent space to generate the final image. Figure 2.9 depicts a diagram of the StableDiffusion architecture.

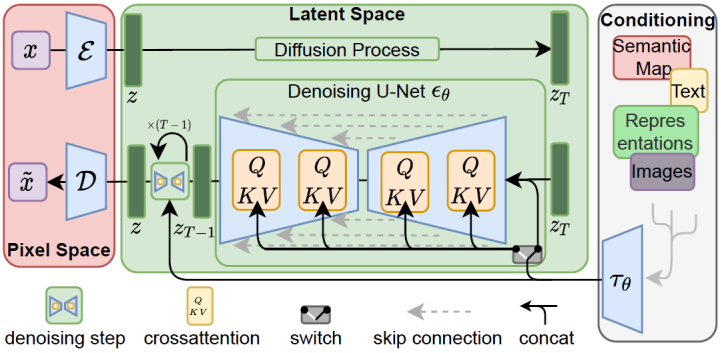


Figure 2.9: *Overview of the architecture of StableDiffusion (Rombach et al. 2021).*

2.3.3 Datasets and evaluation metrics

Datasets. Even though in deep learning good datasets are always needed for achieving a good performing model, it is especially true in this case (Han et al. 2019). Table 2.3 shows a list of the most commonly used inpainting datasets along with the category type, which defines the purpose of the inpainting model.

Dataset	Category Type	No. images	Resolution
ImageNet (Russakovsky et al. 2014)	Objects	14,197,122	Variable
CelebA (Liu et al. 2014)	Faces	202,599	178x218
CelebA-HQ (Karras et al. 2017)	Faces	30,000	1024x1024
Paris Streetview (Doersch et al. 2012)	Scenes / Outdoor	15,000	936x537
Places (Zhou et al. 2016)	Scenes / Outdoor	10,000,000	Variable
DTD (Cimpoi et al. 2013)	Textures	5,640	Variable
NVIDIA Irregular Mask (Liu et al. 2018)	Masks	67,116	512x512

Table 2.3: *Popular public datasets for inpainting*

Evaluation metrics. There is no protocol or unified standard to evaluate inpainting algorithms (Jam et al. 2020). Additionally, often metrics alone are not sufficient and should be combined with a qualitative evaluation (Xiang et al. 2023). However, common metrics are MSE and SSIM, which

are used to measure the quality of the reconstruction (Xiang et al. 2023), and FID to measure the quality of generated samples in the case of GANs (Xiang et al. 2023) and diffusion-based models (Parida et al. 2023).

Mean Normal Squared Error (NMSE) (Xiang et al. 2023). This metric measures the average squared difference between the estimate x and the ground truth y. When used in practice it is normalised. Formally it is defined as follows:

$$NMSE = \frac{||x - y||_2^2}{||y||_2^2} \tag{2.3}$$

Structural Similarity Index (SSIM) (Xiang et al. 2023). SSIM compares the network estimate x and the ground-truth y based on luminance l, contrast c and structure s. SSIM is a weighted combination of these three measurements that estimate a change in structural information. Formally given by:

$$SSIM = [l(x, y)^{\alpha}[c(x, y)]^{\beta}[s(x, y)]^{\gamma}] \tag{2.4}$$

Frèchet inception distance (FID). FID calculates the distance of two distributions. In practice, it is used for comparing the distribution of the estimates e of generative models to the distribution of the ground truth g and is defined as follows:

$$FID = ||\mu_g - \mu_e||_2^2 + Tr(\sigma_g + \sigma_e - 2(\sigma_g\sigma_e)^{\frac{1}{2}}) \tag{2.5}$$

The metric uses the pre-trained Inception-v3 network, in particular, $X_e - N(\mu_e, \sigma_e)$ and $X_g - N(\mu_g, \sigma_g)$ are its activations of layer pool-3 for the estimate and the ground truth. A lower FID is better since denotes a lower distance between the estimate and the ground truth.

3 Analysis and Requirements

As seen in Section 2.1, research on 3D reconstruction has explored methodologies for retrieving information from single or multiple images and utilising them for generating a 3D shape. New methods, based on deep learning, have decreased the assumptions needed to successfully generate 3D geometries. For example, deep learning methods generate 3D shapes from a single or few images of an object, without the necessity of knowing intrinsic camera parameters. Additionally, the object does not need to be in a determined pose or under specific lighting conditions. The latest research focuses on the type of 3D representation to use, how to optimise existing methods and how to achieve a higher level of detail in the output.

However, there are still limitations in the flexibility of these approaches. For example, most state-of-the-art approaches struggle with images of multiple objects in a cluttered environment (Han et al. 2019). Additionally, most of the objects these models are trained with and evaluated on depict fully-visible objects and only a few images have inter-occluded objects.

This chapter explores the problem of occlusion and the impact of the most commonly used dataset, ShapeNet (Chang et al. 2015), on the efficacy of deep learning methods.

3.1 Inter-object occlusion

Self-occlusion and inter-occlusion. The concept of occlusion has been studied in areas such as face recognition (Zeng et al. 2021) and object tracking (Lee et al. 2014). Lee et al. (2014) categorises occlusion in *non-occlusion*, *full-occlusion* and *partial occlusion*, which is further subdivided into *self-occlusion* and *inter-object occlusion*.

Self-occlusion arises when a part of the subject occludes another and, in the case of single image 3D reconstruction, it is omnipresent. In general, 3D reconstruction is ill-posed due to mainly self-occlusion, since there are multiple plausible reconstructions of the non-visible areas (Han et al. 2019). The other type of partial occlusion, inter-object occlusion, occurs when a portion of the subject is occluded by another entity such as an object, person or foreground (Lee et al. 2014).

Since the project focuses on single image 3D reconstruction, the case of full occlusion caused by inter-object occlusion is excluded. Similarly, non-occlusion is also an excluded scenario for single-image 3D reconstruction.

While deep learning approaches tackled the issue of self-occlusion, by achieving 3D reconstruction from a single or few images, there is almost no research that addresses the problem of inter-object occlusion.

3.1.1 Implications of inter-object occlusion.

Dealing with multiple objects. Inter-object occlusion further complicates the problem of 3D reconstruction by introducing at least one additional entity, for example, another object, in the scene. Hence, it transforms the issue of single object 3D reconstruction, to 3D reconstruction of multiple 3D shapes from an image of multiple objects. Dahnert et al. (2021) reconstruct directly

Figure 3.1: *Some sample images from the ShapeNet dataset (Chang et al. 2015; Fu et al. 2021).*

the 3D scene from a single image. Gkioxari et al. (2019), instead, first mask out the furniture in the scene, then individually reconstruct the segmented objects.

Reconstructing an entire scene differs from reconstructing multiple objects. A 3D scene, in addition to any object in the image, would also reconstruct the background, for example, the walls if the image is indoors. Also, the output is a single 3D shape. When reconstructing multiple objects, the background is not reconstructed and not necessarily all of the present objects are reconstructed. Hence, the output is one or multiple 3D shapes of the selected objects.

Object complexity. One major challenge in 3D reconstruction is to be able to capture and reconstruct the complexity of a single object. If fine-grained details occupy a small portion of the overall object, the reconstruction tends to have a good overall score (Fu et al. 2021). However, as the details become predominant with respect to the structure, the reconstruction results tend to be poorer (Fu et al. 2021). Obviously, adding inter-object occlusion increases the difficulty of reconstructing a detailed shape since the model will need to hallucinate the missing details.

An additional challenge is to learn the inter-class similarities while preserving the intra-class distinct features Fu et al. (2021). Inter-class occlusion further complicates this task since there is the risk to occlude key features for distinguishing and determining the type and structure of the object.

3.2 Need for partially occluded images

While there are image-based datasets that include occlusion, such as ImageNet (Russakovsky et al. 2014) and COCO Lin et al. (2015), which are used, for example, for object recognition and classification, there is a lack of occluded datasets for the training of 3D reconstruction models. Since the problem of inter-object occlusion has not been explored, there is a need to gather/generate new training and evaluation data.

Issues with synthetic datasets. ShapeNet (Chang et al. 2015) is the most commonly used synthetic dataset in the field of 3D reconstruction. It offers a large collection of paired images to ground truth shapes of generic objects over 55 different classes. Hence, ShapeNet, which is divided into pre-defined training, validation and test set, has been widely used for training purposes (Fu et al. 2021). Figure 3.1 shows some sample images from the ShapeNet dataset (Fu

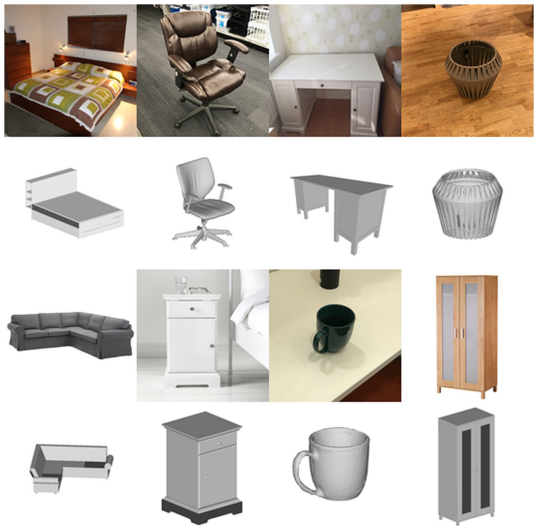


Figure 3.2: *Pix3D samples images with the relative ground truth shapes (Sun et al. 2018; Fu et al. 2021).*

et al. 2021). However, the image collection of ShapeNet and other synthetic datasets (Wu et al. 2014), do not reflect the complexity of real-world images, such as cluttered background, multiple classes of objects, occlusion and lighting variations (Han et al. 2019; Fu et al. 2021). Having a uniform background instead of a cluttered/textured background is not generally an issue. Indeed, a common pre-processing step is to mask out the background, and real-world datasets themselves provide the segmentation mask of the object (Ben-Shabat et al. 2020; Sun et al. 2018). However, multiple classes of objects, occlusion and lighting variation greatly influence the input image, which makes it difficult to perform 3D reconstruction of objects on real-world images after being trained on a synthetic dataset (Fu et al. 2021).

Issues with real-world datasets. There are, actually, some datasets that are paired with 3D ground truth and have at least some images of partially occluded objects. Examples are Pix3D (Sun et al. 2018), IKEA (Ben-Shabat et al. 2020) and PASCAL3D+Xiang et al. (2014), however, they suffer many issues, making them not suitable for training. In the case of the IKEA dataset, it is simply too small and has too few classes compared to other training datasets used in tasks such as object classification and recognition (Han et al. 2019). Pix3D is an extension of the IKEA dataset, however, it still has not enough images and 3D shape pairs to be used as training datasets. Hence, they are used as evaluation datasets. Figure 3.2 shows some sample pairs from the Pix3D dataset (Fu et al. 2021).

PASCAL3D+ is instead a relatively large dataset with 12 categories and about 36 thousand image and CAD model pairs. However, the images are paired with almost generic CAD models. Figure 3.3 (Fu et al. 2021) shows some sample pairs from the PASCAL3D+ dataset, and helps better explain the issue. There are some accurate pairs, in which the CAD model is very similar to the object depicted in the image, or vice versa. However, there are also cases of pairs in which the CAD model is a generic model representing the class of the object in the image. Clear examples are the bus and car models.

When analysing the pairs in detail, it can be noted that also the office chair and the canoe image differ slightly from the relative 3D ground truths. It has to be noted that such a level of accuracy in matching pairs of images to ground truth 3D shapes is not only an issue with PASCAL3D+, but it is something that afflicts, at one level or another, all the 3D reconstruction datasets that provide a paired 3D ground truth. However, this lack of fine-grained details in the training data could lead to increased difficulty in generating fine-grained details.

Figure 3.3: *Some PASCAL3D+ samples images with the relative ground truth CAD models (Xiang et al. 2014; Fu et al. 2021).*

4 Design

This chapter gives an overview on the approaches used for generating a new occluded dataset, e.g., by modifying already existing data as seen in Section 2.2. Additionally, this chapter covers the proposed solutions for improving 3D reconstruction under occlusion, which use fine-tuned models on occluded images (Section 4.1.1), on inpainted images (Section 4.1.2) and both types of images (Section 4.1.3).

4.1 Proposed solutions

4.1.1 Single-view 3D reconstruction on occluded images

Since this is a research project, a lot of time was spent exploring methodologies that resulted in failure, for example, fine-tuning an inpainting model (see Section 5.2.1), or that simply were no longer feasible due to their many issues, which are discussed in Section 4.2.2. The simple idea is to fine-tune the 3D reconstruction model on occluded images. Fine-tuning modifies the weights of the baseline model by exposing them to occluded images, to create a more robust and accurate model.

Therefore, a straightforward method is suggested: using the fine-tuned models directly on occluded images. This method act as a baseline and be compared to the application of inpainting and the combined approaches.

4.1.2 Single-view 3D reconstruction using inpainting

The partially occluded images can be inpainted to hallucinate the missing parts. The idea is that this will compensate the lost information due to occlusion and improve the 3D reconstruction results. The proposed pipline involves pairing the input images with the relative erasing masks, inpaint them, and then fed the inpainted images to the 3D reconstruction network. The pipeline is represented in Figure 4.1.

Inpainting as a pre-processing step. Similar to the intermediate voxel decodings seen in Section 2.1.3, the proposed approach aims to leverage additional information that is extracted from the input image. However, there is a fundamental difference: in our case, the inpainting phase is a pre-processing step of the input, executed by another model, before feeding it to the 3D reconstruction model. In other words, the 3D reconstruction network has not been modified to add an inpainting module. Instead, approaches such as MarrNet (Wu et al. 2017), use intermediate encodings (Section 2.1.1). Hence, have two encoders, a first encoder-decoder to generate the intermediate state, and the second to encode such state into latent space.

The difference may seem subtle in theory, but in practice has various consequences. For example, if the models were combined or in the case of intermediate decodings, one can choose to fine-tune the intermediate encoder-decoder, the 3D encoder-decoder, or both at the same time. In our case, since our models are separate, fine-tuning the 3D reconstruction model will not affect the weights of the inpainting model.

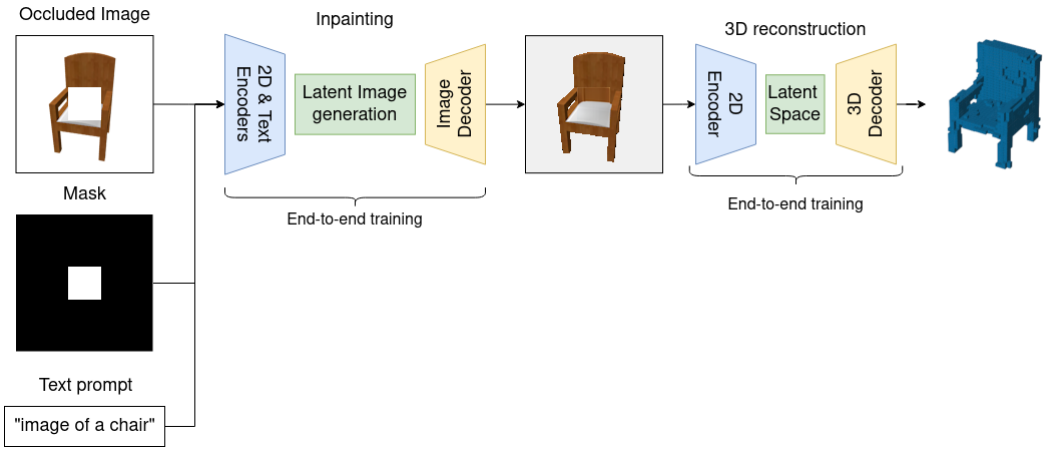


Figure 4.1: *Pipeline for single-view 3D reconstruction in which the input, an occluded image, is inpainted before being processed by a 3D reconstruction model.*

Fine-tuning both the inpainting algorithm and the 3D reconstruction model at the same time could have had the potential of improving results. The idea is that during the training phase, both models' weights will be adjusted to better work together, e.g., with the inpainting module inpainting images to specifically facilitate the 3D reconstruction module. However, this has not been tried and the proposed pipeline has been implemented with the two models separated. The advantages are modularity and simplicity. Indeed, each stage of the pipeline can be treated as a black box and issues are easier to confine and detect. Furthermore, it is easier to determine how the various changes and decisions impact the single models.

Missing masking step. As seen in Section 3.1, in the case of images of multiple objects, masking out the relevant objects and then generating the relative 3D shape is a viable solution. However, the masking step is a solution for any multi-object images, independent of whether such objects were affected by inter-occlusion or not. Hence, the masking step has been left for further work and preferred to focus on the impact of inpainting an addition for improving 3D reconstruction. The proposed pipeline assumes that the input images have already been masked out and the mask is available.

4.1.3 Multi-view 3D reconstruction combining inpainting and occluded images

A variation of the above pipeline is to treat the inpainted result and the original image as two different input images for a multi-view 3D reconstructing algorithm. Expanding on the discussion in Section 4.1.2, If the inpainting step were an intermediate step instead of a pre-processing step, the pipeline could have employed a single-view 3D reconstructing algorithm. The core idea is that the inpainting output could contain defects or have modified the original image structure which are not present in the original occluded image. In fact, feeding the original image could improve the final 3D output since multi-view 3D reconstruction models have a module that blends the input images by retaining the best information of each. Examples are the work of Choy et al. (2016) and Xie et al. (2019).

4.2 Generating an occluded dataset

Pairing real-world 2D images with the relative 3D ground truth shapes is a challenging task. Gathering a collection of 2D images or 3D shapes is not an issue by itself. However, starting

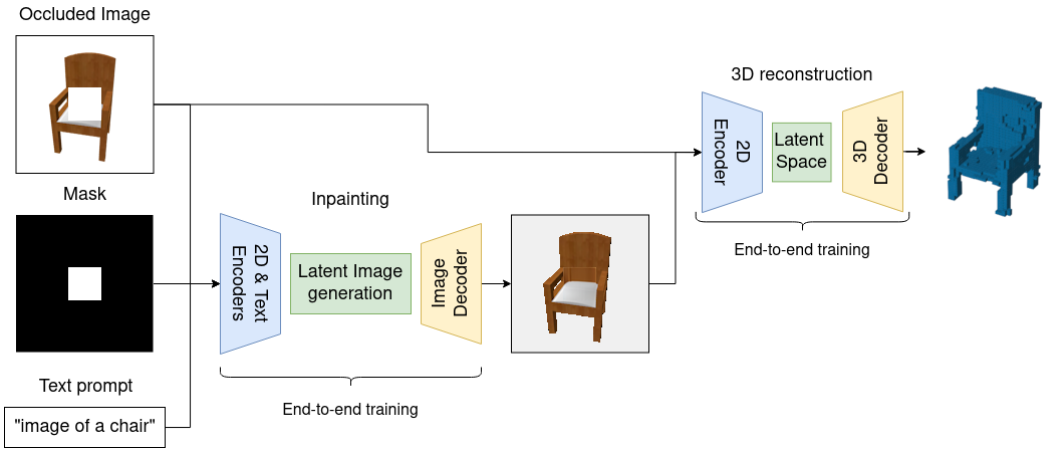


Figure 4.2: *Pipeline for multi-view 3D reconstruction in which both the occluded and inpainted images are used as inputs.*

with a large collection of 3D objects, it is easier to pair them with the relative renders, instead of finding the corresponding real-world objects and taking pictures of them.

On the other hand, retrieving a collection of 3D shapes given a collection of 2D images is the aim of 3D reconstruction. Datasets such as Sun et al. (2018) and Ben-Shabat et al. (2020) were able to gather such datasets, but they are not vast and cover mostly furniture. Data augmentation is a simple and lightweight approach to enrich the dataset. However, since it does not create completely new pairs, it is similarity-preserving (Han et al. 2019).

4.2.1 How to apply occlusion.

In Section 2.2, the data augmentation methods of Random Erasing (Zhong et al. 2017) and Cutout (DeVries and Taylor 2017a) were introduced. Both of these methods have been compared to dropout, which involves randomly deactivating a percentage of neurons at each iteration (Srivastava et al. 2014). The main difference is that Random Erasing and Cutout are applied to the input layer, while dropout is applied to hidden layers.

The consequences are not minor, since by removing visual features or objects at the input layer, they are consequently removed from all the future feature maps. On the other hand, dropout variants are generally applied to each feature map, hence a randomly removed feature could be still present in other feature maps (DeVries and Taylor 2017a).

Additionally, standard dropout is very effective in regularising fully-connected layers but is less powerful when used on CNNs (DeVries and Taylor 2017a). One reason could be that neighbouring pixels in an image tend to share similar information, hence if any of them are dropped out, their information is likely to be preserved anyway. Instead, by removing a continuous region of the image, it is more likely that such information will be lost (DeVries and Taylor 2017a). This implies that with dropout, the model increases robustness to noisy inputs, while methods such as Random Erasing and Cutout increase robustness to partial occlusion. Indeed, the resulting layer better resembles occluded inputs and incentivises the network to consider more of the image context (DeVries and Taylor 2017a).

To summarise, dropout has the advantage of being implemented without generating a new dataset, however, it is not fit for simulating partial occlusion when using a CNN. Instead, Random Erasing and Cutout are data augmentations techniques more suitable for this task.

4.2.2 Generating realistic synthetic images

GAN vs Diffusion models. As seen in Section 2.3, inpainting deep learning approaches are better suited than traditional inpainting methods to recreate complex scenes or inpainting large sections. Of the deep learning methods, the most effective and commonly used architectures are GANs and Diffusion-based models. For this project, diffusion-based models have been chosen since are easier to train compared to GAN and the better capabilities of generalisation (Dhariwal and Nichol 2021).

In the next section, there will be an explanation of the use of *StableDiffusion* (SD) (Rombach et al. 2021), a latent-diffusion model for image inpainting. However, StableDiffusion can be used for multiple tasks such as generating new images or image-to-image (img2img) translation. Hence, due to the lack of datasets with real images, or lack of realism in synthetic datasets, it has been conducted an exploration of the uses of StableDiffusion for the creation of a more realistic version of the ShapeNet-Chair dataset.

Img2Img and Dreambooth. The StableDiffusion Img2Img model generates new images based on an input image and a text prompt. The text prompt can be used to further describe and explain the content of the generated image. The idea consists to generate realistic images of the chair models in the ShapeNet-Chair dataset in different angles and settings. Then, pair such generated images to the already existent voxel ground truth, thus creating a dataset usable for the supervised training of 3D reconstruction algorithms. However, qualitative experiments resulted in either high-fidelity images that lacked realism or the opposite. Additionally, specifying to generate top or back views led to qualitatively poor results.

In an attempt to solve these issues, we paired StableDiffusion with *Dreambooth* (Ruiz et al. 2022). Dreambooth is a fine-tuning diffusion model framework, that allows the generation of images of a specific subject when a unique text identifier is added to the text prompt. For example, given a collection of a few images (typically 3 to 5) of a subject (i.e., of a specific cat) and the relative class name (i.e., "cat"), the diffusion model is fine-tuned to encode a unique text identifier, [v], that corresponds to the subject (Ruiz et al. 2022). Thus, a text prompt containing the unique identifier plus the class name will generate images about the specific subject. In our experiment, SD was fine-tuned with three images of different views of a specific chair model, paired with the identifier "chair". Hence, by generating an image with the text prompt "A realistic [v] chair in a living room", a chair similar to our chair model was generated. While some of the outputs were realistic images coherent with the chosen chair (see Figure A.2), part of the created images lacked consistency and had visual defects. Additionally, instructing to generate top or back views continued to generate qualitatively poor results. While there could have been a model trained to score and filter the created images, similar to what a discriminator would do in a GAN architecture, the implementation of such a model/pipeline would have been time-consuming for implementing and training it. Indeed, the high inference time does not lend itself to using this method for the creation of a large dataset. Due to the generation of a realistic dataset not being the goal of the project, this approach was rejected.

5 Implementation

This chapter covers the implementation of the proposed approaches in the previous chapter and the generation of a set of occluded variants of the ShapeNet (Chang et al. 2015) dataset.

5.1 Occluded dataset generation

In this section, there will be discussed the augmentation of the already existing ShapeNet dataset (Chang et al. 2015) through the use of a novel approach that combines Random Erasing (Zhong et al. 2017) and Cutout (DeVries and Taylor 2017a). Then, the different mask hyper-parameters will be discussed. Finally, the generated datasets, that will be used as evaluation datasets, are listed in Section 5.1.2.

5.1.1 Partially occluded ShapeNet

ShapeNet-Chairs. As seen previously, ShapeNet (Chang et al. 2015) contains 55 classes and more than 51,000 images of objects paired with the relative 3D voxel grid. However, due to the limited hardware resources using such a big dataset was not feasible for this project. Hence, only a subset of it, the images of the chair class, are used. The choice of using the class chair specifically is because it is a class common to all the other cited datasets (Sun et al. 2018; Ben-Shabat et al. 2020; Xiang et al. 2014), but also to the ImageNet dataset (Russakovsky et al. 2014) which is used for inpainting. Additionally, chairs are supported by HM3D-ABO (Yang et al. 2022b): a pipeline for creating photo-realistic datasets. Hence, a fine-tuned model could have been evaluated on it. While this was an argument that, at the early stage of the project, further led to choosing the chair subset of ShapeNet, in the final stage of the project, HM3D-ABO has not been further explored due to time constraints. Thus, for the above reasons, the project used the chair subset of ShapeNet, which will be referred to as ShapeNet-Chair.

A method for applying occlusion. Random Erasing (Zhong et al. 2017) and Cutout (DeVries and Taylor 2017a) are both very similar and suitable for the task of simulating partial occlusion. Our approach to generating an occluded ShapeNet-Chair combines these two methods and builds on them. Mask hyper-parameters greatly influence the final result, hence, we generated a series of occluded ShapeNet-Chairs datasets to evaluate how different mask hyper-parameters affect the information loss. Indeed, a method to partially occlude objects should allow quantifying and controlling the loss of information caused by the generated occlusion. This is essential for evaluating the difficulty of the task, both for the inpainting and the 3D reconstruction model, and for comparing the results obtained.

Xiang et al. (2023), when evaluating several inpainting algorithms, uses a centre mask of 128×128 pixels for images of 256×256 pixels and a mask of 64×64 for images of 128×128 pixels. This causes a loss of information of 25% since every pixel of the image is valuable information. Indeed, the inpainting algorithm has to reconstruct everything of the masked out 25%, independently if the pixel represents the main subject or not. In the case of ShapeNet, the image is composed solely of a uniform background and the object, which is placed at the centre of the image. Hence, all the information is in the centre pixels that represent the object, which is a small portion of the

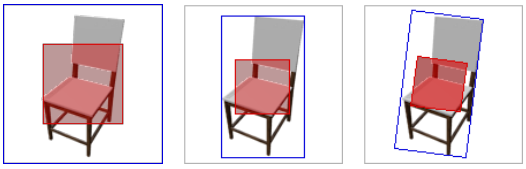
entire image. If a centre mask that covers 25% of a ShapeNet image is applied, it should cover most of the pixels representing the object, resulting in an information loss higher than 25%.

Cropping and resizing ShapeNet. One approach to lower the information loss is to increase the pixel area covered by each object in the images. Thus, each image in ShapeNet-Chair has been cropped by removing the edges outside of the bounding box of the object and then resized to match the original image size. The resulting image had the advantage of increasing the object size, increasing the portion of pixels that conveyed meaningful information. However, the major downside was to reduce the dataset variety: renders that depicted the same object from a similar angle but had a different camera distance, resulted in very similar images after applying cropping and resizing (see examples in Figure A.7).

IRE. Data image erasing augmentation methods apply masks at random locations, instead of them being fixed at the center (Zhong et al. 2017; DeVries and Taylor 2017a). In particular, (Zhong et al. 2017) applies Random Erasing using different schemes, including IRE (Image-aware Random Erasing) and ORE (Object-aware Random Erasing). Applying the erasing mask at random locations, instead of applying it to the centre, should lower the amount of information loss since there are higher chances for the masks to cover the background and not the object.

ORE. ORE is another scheme proposed by (Zhong et al. 2017), which calculates the bounding box of the object and then applies random erasing into its area. Similarly to cropping and resizing the images, this approach allows to exclude most of the background. Additionally, it removes the percentage of pixels based on the bounding box size, instead of the image size. Additionally, to evaluate the information loss of applying ORE at random locations, we propose a variant that places the mask at the centre of the bounding box.

MORE. To try to achieve even higher precision of information loss, we propose a new approach: Minimum Object-Aware Random Erasing (MORE). The proposed approach allows rotation and hence calculates the minimum bounding box of the object. The erasing mask is then applied to the center of the minimum bounding box. Figure 5.1 shows the effects of applying the IRE, ORE and MORE schemes.



(a) IRE *(b) ORE* *(c) MORE*

Figure 5.1: *Types schemes for applying erasing: (a) IRE (Zhong et al. 2017), (b) ORE (Zhong et al. 2017), (c) MORE. Blue lines represent the bounding boxes, while the red rectangles are the area on which erasing has been applied.*

Additional mask hyperparameters. Other mask hyper-parameters need to be set before being able to apply erasing as listed below.

- **Location**: (Zhong et al. 2017) IRE and ORE schemas apply the mask randomly, while DeVries and Taylor (2017a) experiment with applying the mask at the centre as well. We generate different datasets, experimenting with both random and centre locations.
- **Erasing area percentage**: is the percentage of the image that should be erased. It could be a fixed value or a random value over an interval. We used a random value during fine-tuning of the 3D reconstruction algorithm (Section 5.3). For all the other instances, a fixed value has been used.
- **Shape**: DeVries and Taylor (2017a) showed that choosing a rectangular or squared mask shape does not have a huge impact on performance. In general, we implemented the

algorithm to use a squared mask. However, most objects' bounding boxes are rectangular and sometimes their width (or height) is not large enough to generate squared masks that cover the requested erasing area. In such cases, a rectangular shape is generated.

- **Erasing probability**: for evaluating datasets, the probability of applying erasing has been set to 1. Instead, we can explore the impact of choosing different probabilities when fine-tuning the 3D reconstruction algorithm.

5.1.2 List and explanation of generated datasets

The novel methods applies a zero mask using the IRE and ORE schemes proposed by (Zhong et al. 2017) plus a novel scheme, MORE. For the IRE and ORE schemes, the mask is either applied at random or at the centre, while MORE applies the mask only at the center. The erasing area percentage could be fixed or random over an interval. The dimensions of the mask are not fixed, but they are generated depending on the scheme used and the specified erasing area. Hence, its shape is a rectangular and specifically a square if the dimensions of the bounding boxes allows. Finally, an erasing propability parameter can be set to determine how often to apply such erasing to the given images.

Table 5.1 shows the list of generated datasets by combining the mask hyper-parameters discussed above. The dataset name is given by the concatenation of the schema applied, the mask location and the mask size.

The purpose of the *IRE datasets* is to evaluate the information loss caused by IRE when applied at the centre and random locations with different erasing area percentages. *ORE datasets* have been generated to evaluate the proposed pipelines and fine-tuned models. Additionally, it is evaluated the information loss caused by ORE when applied at the Center and Random location using different erasing sizes. *MORE datasets* are used to evaluate the impact of using a minimum bounding box instead of a bounding box when calculating the mask dimensions, given an erasing area percentage and has only been applied at the centre.

Dataset	Mask size	Mask scheme	Mask position	Mask shape
IRE_Center_15	15%	IRE	Center	Rectangular
IRE_Center_25	25%			
IRE_Center_50	50%			
ORE_Center_15	15%	ORE	Center	Rectangular
ORE_Center_25	25%			
ORE_Center_50	50%			
ORE_Random_15	15%	ORE	Random	Rectangular
ORE_Random_25	25%			
ORE_Random_50	50%			
MORE_Center_15	15%	MORE	Center	Rectangular
MORE_Center_25	25%			
MORE_Center_50	50%			

Table 5.1: *The list of the generated occluded variations of the ShapeNet-Chairs dataset.*

5.2 The inpainting step

For implementing the inpainting step, two inpainting diffusion models have been explored: *Palette* (Saharia et al. 2021) and *StableDiffusion* (Rombach et al. 2021). Palette required fine-tuning to be applied for inpainting the ShapeNet (Chang et al. 2015) dataset, which was unsuccessfull. StableDiffusion, being trained on the ImageNet (Russakovsky et al. 2014) dataset, did not need

fine-tuning. Hence, it will be covered the process of selecting its hyperparameters for effectively applying inpainting in our case.

5.2.1 Palette

Palette (Saharia et al. 2021) is a diffusion-based model that performs colourisation, inpainting, JPEG restoration and uncropping. The peculiarity of Palette is being a generalist model, which can perform all the above tasks without the need for specific fine-tuning or customisation, while still achieving competitive performances with state-of-the-art models. The initial idea was to use this flexibility to expand the type of input images. For example, the pipeline could also support cropped images of chairs, without the need to change the inpainting algorithm or implement a new one and still reach competitive results. A diagram showing the inpainting process of Palette is shown in Figure 5.2.

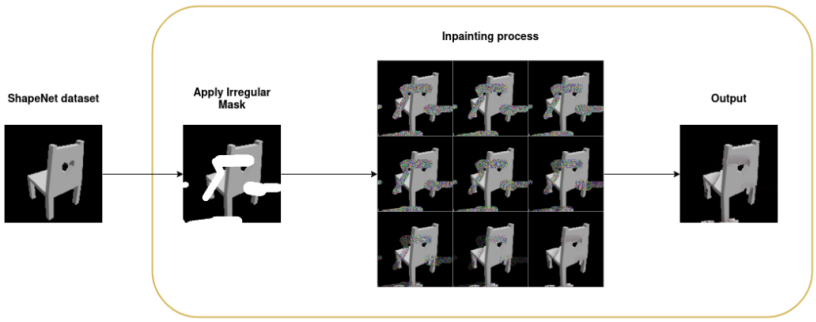


Figure 5.2: *The process of inpainting of Palette on the ShapeNet-Chair dataset. An irregular mask is first applied to cause information loss that the algorithm has to fill. The iterative inpainting process is typical of a diffusion-based algorithm.*

Fine-tuning and training of Palette. For the scope of this project, a checkpoint trained on ImageNet would have been ideal, due to the high number of different classes and the numerous images contained, which could have led to a better generalisation for object inpainting. While Palette has been trained on ImageNet, there is not an official release of its pre-trained models. However, an unofficial implementation of Palette[1] offers two checkpoints of such implementation trained respectively on the CelebA-HQ dataset and Place2 Dataset. Since neither the official version nor this unofficial implementation released a pre-trained model on ImageNet, the Places2 checkpoint has been used as a base for transfer learning. Indeed, even if Places2 contains images of indoors/outdoors which have little in common with full images of single objects, the idea is to leverage the already learned ability to extract general features, such as edges and shapes and apply it to the specific case of chairs.

However, fine-tuning results were far from satisfactory both qualitatively and quantitatively. The training loss and the validation loss neither increased nor decreased, suggesting that the task was too complex for the model to learn. Attempts to use grey images and decrease the mask size also were unsuccessful. The training loss decreased but the validation loss did not improve, suggesting that the subset on which the model was trained was too small and not representative.

5.2.2 StableDiffusion

The release of the pre-trained weights of StableDiffusion (SD) Rombach et al. (2021) has been a game-changer for individuals, privates or researchers, who needed a versatile and effective

[1] github.com/Janspiry/Palette-Image-to-Image-Diffusion-Models

inpainting model, without the resources for training or fine-tuning a new one. Indeed, SD has been trained on various datasets that focus on different subjects such as ImageNet Russakovsky et al. (2014) a generalistic dataset that contains over 14 million images, Place2 and CelebA.

As explained in Section 2.3.2, SD allows to be conditioned with additional different inputs and used in multiple tasks such as super-resolution and inpainting. Contrary to Palette (Saharia et al. 2021), which is a generalist model, StableDifussion has a specific fine-tuned checkpoint for each task. For example, in this project, it is used the StableDiffusion v1.5 checkpoint for inpainting[2]. StableDiffusion can run on a local server which provides a simple yet comprehensive user interface (UI), or an API[3]. First and qualitative experiments were conducted using the user interface, while the generation of the inpainted datasets has been performed using the API.

Text prompt engineering. The generated inpainting content can be conditioned by a *text prompt* and a *negative text prompt*. The text prompt could be a general description with some specific details, of the desired content. It highly influences the generation process, as much as the input image. On the other hand, the negative text prompt is optional and indicates what should not be included in the final image. Longer, more descriptive prompts usually lead to better results, however, they should be general enough to be applied to any chair image.

Qualitative experiments were performed using short text prompts such as "render of a chair" and longer prompts such as "rendering of (one) 3D (chair), render, hyper-realistic". Additionally, keywords could have been put between parenthesis to explicitly give them more weight compared to other words. When inpainting images with a high occlusion ratio (i.e., >50%) only a few distant parts of the chair were visible. The resulting images often were multiple chairs or a chair and another object. We added "(one) chair" or even "(one) singular chair" in the text prompt, but did not help. The text prompt "one chair" was also given to generate a completely new image, which resulted in one image containing two chairs. It seems that the concept of quantity has not been synthesised by SD.

One additional issue was the sporadic generation of text in the masked region. The solution, fast and effective, was to include "text" in the negative text prompt. Figure 5.3 shows a sample input used for the inpainting process.

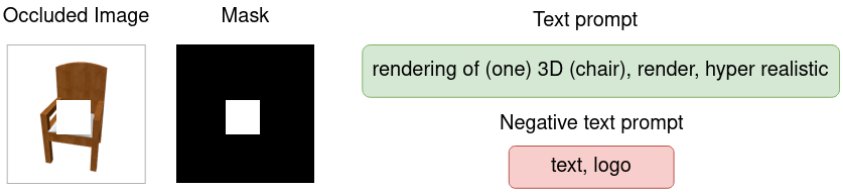


Figure 5.3: *Input used by StableDiffusion for inpainting, comprised of an occluded image, an inpainting mask, a text prompt and the optional negative prompt.*

Selecting the inpaint area. There is the option to use the *whole image* or *only masked region* for the inpainting process. By selecting the whole image option, the entire image will be used for the inpainting process, influencing the generated content and increasing the inference time. On the other hand, using only the masked region, the neighbouring pixels will not influence the inpainting result at all and there is a little gain in performance. Experiments using only the masked region option resulted in two nested images of chairs: the original chair and another newly generated chair in the masked region (see Figure A.4). It is possible to add padding to the masked region, the *only masked padding*, to include the neighbouring pixels in the inpainting process.

[2]The checkpoint can be downloaded from huggingface.co/runwayml/stable-diffusion-inpainting
[3]The code for running a local web UI can be found at github.com/AUTOMATIC1111/stable-diffusion-webui

Classifier Free Guidance (CFG). The CFG (Ho and Salimans 2022) value determines how much SD will follow the text prompts to generate the new image. Lower values, such as 0, indicate ignoring the prompt, while higher values, such as 30, indicate closely following the prompt. The default value is 7, which indicates a general balance between sample fidelity and mode coverage. Our qualitative experiments (see Figure A.3) showed that *CFG*=7 is indeed a good balance.

Sampling steps. The diffusion process is an iterative, refining process of removing noise from an input image. The sampling steps parameter defines the amount of iteration that the inpainting model should perform. While there is generally a gain in terms of the quality of the image if a minimum of sampling steps are performed (i.e., 10-20), using a higher number of sampling steps is not always preferable. Indeed, apart from resulting in a higher inference time, there is a threshold, depending on the specific instance, over which there is a decrease in image quality.

Denoising strength and inpaint fill. Similarly to the CFG, this value determines how much SD will follow the input image to generate the new image. The value range is between 0, which will not change the input image, to 1.0, which indicates to change completely the masked region.

The inpaint fill option allows you to select the initial content of the masked area, which can be:

- **fill**: a blurred version of the image;
- **original**: the original content of the input image;
- **latent noise**: random noise;
- **latent nothing**: a uniform filling of the average of the neighbouring colours.

As can be seen from Figure A.6, in this specific application, the denoising strength strictly correlates to the inpaint fill. Indeed, the occluded region is completely black, as the background, and does not hold any information. Hence, it is a trade-off between the inpaint fill option, which determines what type of information to add, and the denoising strength, which determines how much of that information will be relevant. After performing some qualitative experiments, we selected a latent noise inpaint fill with a denoising strength of 0.90.

When applied to the occluded versions of ShapeNet-Chair, Fill and Original inpaint fill options are quite similar. Indeed, the background often covers a great portion of the image, hence the Fill option will be a very dark image, similar to the black values of the original input image. Latent nothing is the most logical choice: it fills the masked area with the average of the colours of the object. However, in practice, it is not always the case. If the erasing region is at the edges of the object silhouette, most of the surrounding pixels regard the background, hence the average will be a dark, uniform colour. A dark uniform colour not only does not add sensible information about the object but also, misdirects the inpainting process, by simulating a hole into the object. On the other hand, a masked region of the colour of the object could suggest that the object does not have a gap, where there could be one. Instead of using the Latent Nothing option, which has a highly variable effect depending on the position of the erasing mask, we use Latent noise. Latent noise acts as a blank slate since there is no unique, predominant value. A denoising strength, higher than the default value of 0.75, such as 0.85-1.0, lowers the influence of the starting content, and hence could be used for mitigating the impact of a uniform filling. However, the pixels surrounding the mask region are valuable information. Hence, instead of just using a denoising strength of 1.0 and anything as masking content, we use a denoising strength of 0.9 with Latent noise.

Image size. StableDiffusion is particularly efficient compared to other diffusion models since generates new images on the latent space level, instead of the pixel space. In particular, images are encoded into a series of patents of size 64×64. For this reason, it is advisable to choose an image size to be a multiple of 64. To achieve even better results, it is best to use the image size on which SD has been trained: 512×512. Higher resolutions increase the memory requirements and the inference time. The multiple of 64 closest to the image size of the ShapeNet images,

which is 137 × 137, is 128 × 128. However, our qualitative experiments, see Figure A.5, show that using lower resolutions than 512 × 512 leads to poor results.

ORE_Random_Inpainting and ORE_Random_Combined datasets. After performing qualitative evaluations, we selected the following hyper-parameters:

- Text prompt = "rendering of (one) 3D (chair), render, hyper-realistic"
- Negative prompt = "text,logo"
- Inpaint area = only masked region
- Only masked padding = 64
- CFG scale = 7
- Sampling steps = 10
- Denoising strength = 0.9
- Inpaint fill = latent noise
- Image size = 512x512

Xie et al. (2019) tested their Pix2Vox model using the first image of each model in the ShapeNet dataset. To have more consistent results, we did the same. Instead of inpainting the entirely the ORE_Random datasets, we just needed to inpainted the first image of each model, generating the ORE_Random_Inpainting_15, ORE_Random_Inpainting_25 and ORE_Random_Inpainting_50 datasets. Additionally, to test the combined approach of using both the occluded and the inpainted image, we combined the ORE_Random and the ORE_Random_Inpainting datasets into ORE_Random_Combined datasets.

5.3 The 3D reconstruction step

Pix2Vox. The Pix2Vox model (Xie et al. 2019) allows both single-image and multi-image 3D reconstruction and hence it is a good choice for implementing the two proposed pipelines. Pix2Vox achieves state-of-the-art performance regarding single-view 3D reconstruction with an MIoU of 0.661 on the ShapeNet dataset for 32x32x32 voxels representation (Fu et al. 2021).

Additionally, its Context-aware Fusion module combines the information of multiple images, allowing for multi-view 3D reconstruction. An overview of the Pix2Vox network used for multi-image 3D reconstruction is shown in Figure 5.4. In particular, the Context-aware module generates a score map for each coarse volume which allows determining and then selecting the highest quality parts, as seen in Figure 5.5.

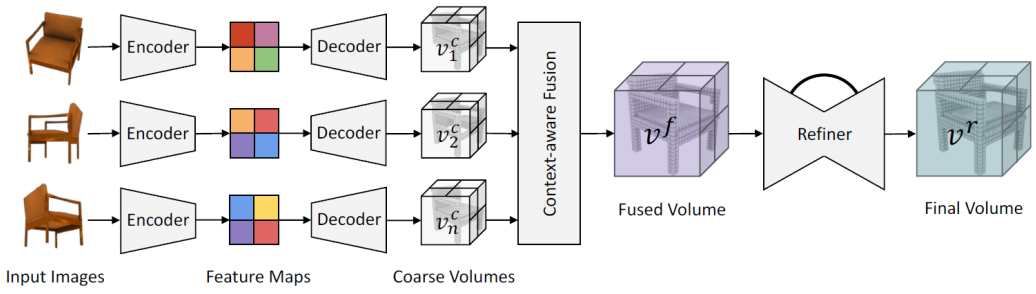


Figure 5.4: *For each input image, the encoder extracts its feature maps, which will be then used by the decoder to retrieve a coarse voxel representation. Since multiple images are provided, the Context-aware Fusion module combines them into a fused volume by selecting the highest quality parts of each coarse volume. Finally, the Refiner module improves the reconstruction of the fused volume into the final voxel. Note that their same encoder-decoder weights are applied to all images (Xie et al. 2019).*

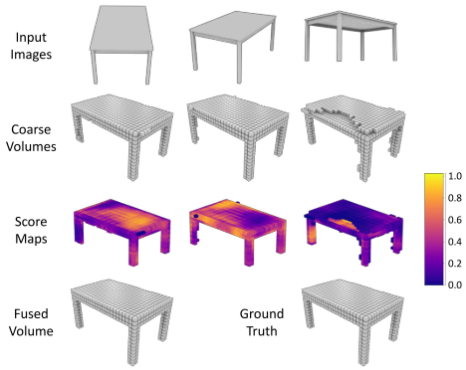


Figure 5.5: *Generated score maps by the Context-aware Fusion module. A higher score implies higher quality (Xie et al. 2019).*

Fine-tuning Pix2Vox Since Pix2Vox supports and has been trained on ShapeNet, the available checkpoints[4] are ready to use in the proposed pipelines. Additionally, to the Pix2Vox baseline, we propose four fine-tuned Pix2Vox checkpoints to compare to the proposed pipelines on the generated occluded ShapeNet-CHair datasets.

When fine-tuning Pix2Vox we selected three hyper-parameters to evaluate: the *learning rate (lr)*, the *erasing probability (p)* and the *mask size*. The learning rate indicates the step size used to adjust the weights concerning the loss gradient. The erasing probability indicates the probability of either applying or not erasing the training image. The mask size, which is an interval and not fixed, indicates the percentage of the image erased. Both erasing probability and mask size further increase the variation of the online generated dataset. Except for the batch size, which had to be lowered from the default 64 to 32 for allowing the training on a GeForce GTX 1060 (6GB), it has been used the default values of all other hyper-parameters.

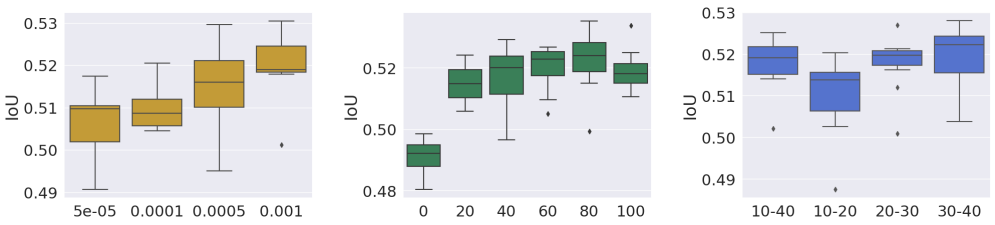
The proposed checkpoints (Erasing_10-20, Erasing_20-30, Erasing_30-40 and Erasing_10-40) were fine-tuned on an occluded version of ShapeNet-Chairs, on which erasing has been applied online at random locations and using the ORE schema (Zhong et al. 2017). A data augmentation method is applied online if applied during training time on random training samples. Using the same approach, an occluded version of ShapeNet-Chairs was also used as the validation dataset, on which the model is evaluated at the end of each training epoch. The erasing probability and mask size used in the validation dataset are fixed at 100% and 10-30% respectively.

It has to be highlighted that the datasets listed in Section 5.1.2 were not used for training or validation. Instead, they were only used for evaluating and comparing the baseline and the single-view and multi-view pipelines, and the fine-tuned models. there have not been used the generated datasets. Applying image erasing at training time, instead of using the previously generated datasets, has the advantage of reducing the overfitting of the model to the specific dataset, since, at each epoch, the erasing mask is applied differently.

Hyper-parameters selection. To select the best values for each of the selected hyper-parameters (learning rate, erasing probability and mask size), when evaluating one of the hyper-parameters, we fixed the remaining two. A summary of the results obtained can be seen in Figure 5.6.

The default *lr*, used for training the baseline, is 0.001. We tried with four values, that start from the default *lr* and gradually decrease: 0.001, 0.0005, 0.0001, 0.00005. Erasing probability was fixed at 100% and erasing area percentage was 10-30%. After the experiments, we fixed $lr = 0.001$ since scored the highest MIoU on the test dataset.

[4] Pix2Vox checkpoints are available on the official repository github.com/hzxie/Pix2Vox



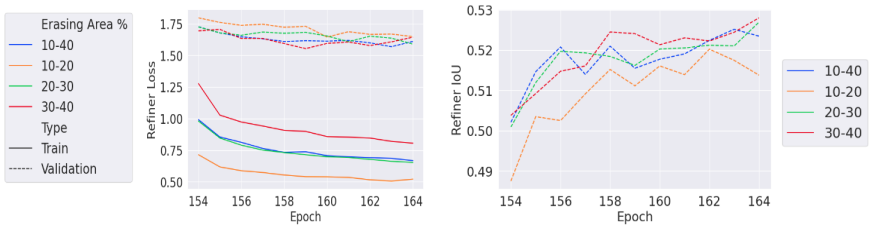
(a) Learning rate *(b) Erasing probability* *(c) Erasing area percentage*

Figure 5.6: *Evaluation of (a) learning rate, (b) erasing probability and (c) mask size hyper-parameters on the validation dataset.*

Then, we tested fine-tuning Pix2Vox with erasing probability of $0, 20, 40, 60, 80, 100$. Evaluating the impact of zero erasing probability is important to discern if the model is improving for the additional iterations or for actually being exposed to an occluded dataset.

Finally, the proposed checkpoints are *Erasing_10-20*, *Erasing_20-30*, *Erasing_30-40* and *Erasing_10-40*. They have been generated by fixing $lr = 0.001$ and $p = 80\%$, and have erasing area percentage between $[10, 20]$, $[20, 30]$, $[30, 40]$ and $[10, 40]$ respectively.

The final fine-tuned checkpoints have been trained for 11 epochs, and their refiner loss and IoU results can be seen in Figure 5.7. As can be seen from the graph of the refiner IoU results, the models generally improve with the increase of iterations. However, the refiner BCE loss graph shows that some models seem overfitting. Indeed, while the training loss always decreases, the test loss increases after a few epochs (i.e., the Erasing_30-40 checkpoint) or remain almost invariant (i.e., the Erasing 10-40 checkpoint). Hence, since the checkpoints have been saved at each epoch, instead of choosing the latest or the best-performing checkpoint (in terms of IoU values), we manually applied *early stopping*. In practice, for each model, we selected the last checkpoint that led to an improvement in the test loss. Thus, we selected the checkpoint at epoch 159 for Erasing_30-40, epoch 156 for Erasing_20-30, epoch 162 for Erasing_10-20 and epoch 158 for Erasing_10-40.



(a) Train and validation Refiner loss at each epoch for each fine-tuned model.

(b) IoU values of the Refiner output 3D voxel grid, compared to the ground truth, at each epoch for each fine-tuned model.

Figure 5.7: *Graph (a) shows the train and validation loss of each fine-tuned model, while graph (b) shows the relative IoU values obtained.*

6 Evaluation

This chapter analyses the information loss, compared to the ground truth images, of the generated IRE, ORE and MORE datasets. Following the generated fine-tuned models are compared to the baseline on the proposed approaches: using only occluded images, applying inpainting and employ both images. Finally, the current limitations and possible solutions that could be implemented in future work are discussed.

6.1 Analysis of the generated datasets

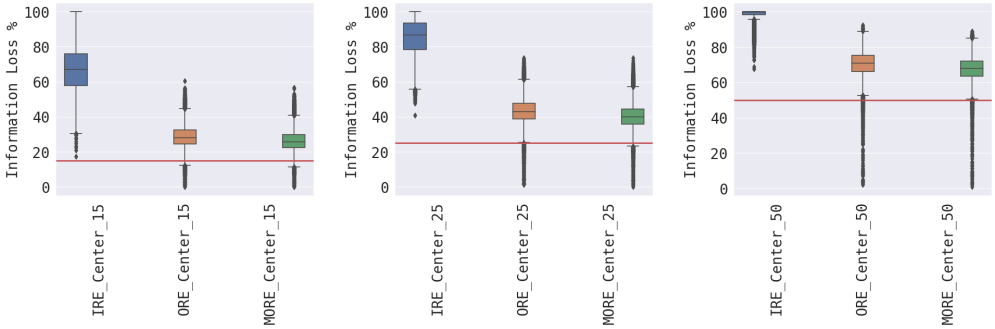
In the Appendix are displayed some samples of the IRE datasets (see Figures A.8 and A.9), the ORE datasets (see Figures A.10 and A.11) and the MORE datasets (see Figure A.12).

Information loss. An analysis of the information loss caused by each erasing approach has been conducted. To calculate the information loss, for each image, it has been calculated the relative percentage difference of the sum of pixels inherent to the object before and after applying erasing.

Figure 6.1 the information loss of the IRE, ORE and MORE datasets with erasing applied at the centre. It can be noted that the medians of all methods are above the target information loss, indicating that they generate occlusions that are more challenging than the wanted level. Across all erasing area percentages, the ORE and MORE methods achieve similar symmetric distributions, with MORE achieving slightly better results. On the other hand, the IRE method performs poorly in all cases. The distribution of IRE_Center_15 has a standard deviation (σ), which indicates the spread of the values around the mean, of 12.8, a lower quartile value of 57.8% and an upper quartile value of 76%. As the erasing area increases, the IRE spread decreases. At 50% of erasing area percentage, IRE_Center_50 has a right-skewed distribution with $\sigma = 3$ and a median of 99.8% of information loss. This means IRE_Center_50 is mainly composed of almost empty images, while IRE_Center_25 applied an occlusion of 78.3% to 93.4% to half of its images. MORE_Center_15, MORE_Center_25 and MORE_Center_50 datasets achieve a mean information loss of 25.6%,39.9% and 67.8% respectively, which is lower than 2.4%, 2.9% and 2.5% respect to the ORE counterparts. Compared to IRE, they have a large number of outliers, especially below the lower quartile, with some samples reaching a 0% information loss. This could be due to chair models having a gap at the centre or an incorrect application of the bounding box.

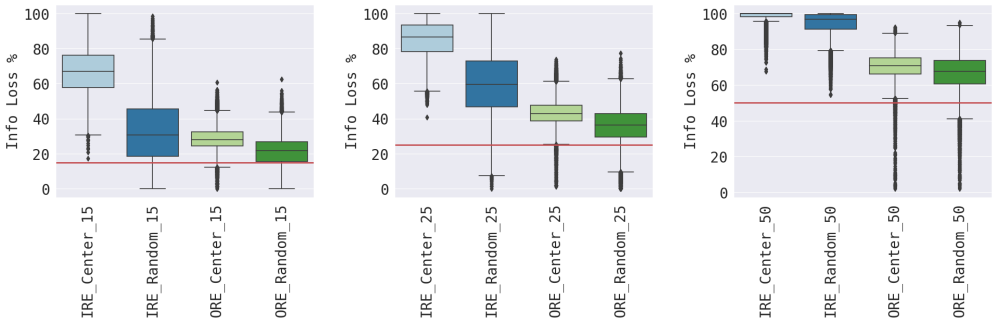
Figure 6.2 shows the information loss of IRE and ORE datasets with masks applied both at the centre and at random. It can be seen that for the erasing area percentages of 15% and 25%, applying erasing at random locations leads to an overall information loss closer to the target, for both IRE and ORE datasets. Applying erasing randomly lower the mean information loss of IRE_Center_15, IRE_Center_25 and IRE_Center_50 by 33.9%, 25.4% and 4% respectively. On the other hand, applying erasing at random lower the mean information loss of ORE_Center_15, ORE_Center_25 and)RE_Center_50 by 7.7%, 7.8% and 3.8% respectively. An expected consequence of applying erasing at random is the increase in standard deviation compared to the centre approach. However, since the upper quantile values are, in the worst case, only slightly higher than the upper quantile values of the centre approach, applying erasing at random still seems the best method.

Choosing ORE_Random datasets The mean information loss of the ORE_Random_15, ORE_Random_25, and ORE_Random_50 datasets is the closest, of all the generated datasets, to the target information loss. Hence we evaluate the proposed solutions on the ORE_Random_15, ORE_Random_25, and ORE_Random_50 datasets.



(a) Target info loss 15% *(b) Target info loss 25%* *(c) Target info loss 50%*

Figure 6.1: *Percentage of information loss of IRE_Center, ORE_Center and MORE_Center datasets grouped by erasing area percentage: (a) 15%, (b) 25% and (c) 50%. The ideal information loss percentage is shown by the red horizontal line.*



(a) Target info loss 15% *(b) Target info loss 25%* *(c) Target info loss 50%*

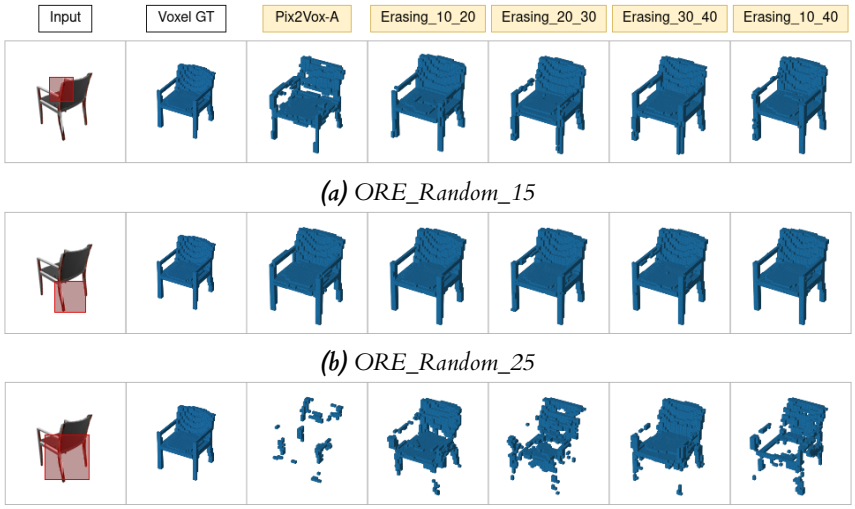
Figure 6.2: *Percentage of information loss of IRE_Center, IRE_Random, ORE_Center and ORE_Random datasets grouped by erasing area percentage: (a) 15%, (b) 25% and (c) 50%. The ideal information loss percentage is shown by the red horizontal line.*

6.2 Analysis of the proposed approaches

6.2.1 3D reconstruction without inpainting

We evaluated the generated fine-tuned models (Erasing_10-20, Erasing_20-30, Erasing_30-40 and Erasing_10-40) and the baseline (Pix2Vox-A) on the ORE_Random datasets without applying inpainting (samples are shown in Figure 6.3). Additionally, we evaluate the checkpoints and the baseline on the vanilla ShapeNet-Chair (Chang et al. 2015) and Pix3D datasets (Sun et al. 2018).

Performance on ORE_Random datasets. As can be seen from Figure 6.4a the proposed fine-tuned models perform better than the baseline. As expected, the overall performance of all checkpoints decreases as the erasing area percentage, and thus the information loss, increases. The baseline generates the lowest mean IoU, which decreases by 16% when incrementing the erasing area from 15% to 25%, and further decreases by 43% when the erasing area is 50%



(a) ORE_Random_15

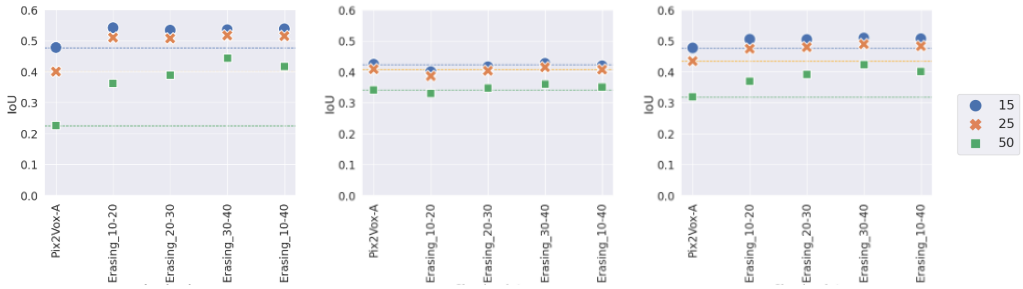
(b) ORE_Random_25

(c) ORE_Random_50

Figure 6.3: *3D voxel reconstruction of the proposed fine-tuned models and the baseline on sample 3a0e392db610f1a1504d5af97121b5f of the (a) ORE_Random_15, (b) ORE_Random_25 and (c) ORE_Random_50 datasets.*

instead of 25%. In comparison, the fine-tuned models lose on average 4% and 21$ of mean IoU loss. On average, the fine-tuned models achieve 0.53, 0.51 and 0.40 mean IoU on the ORE_Random_15, ORE_Random_25 and ORE_Random_50 datasets. Such values are 12%, 27%, and 81% greater than the results of the baseline. In particular, the Erasing_30-40 checkpoints, on ORE_Random_50, achieve an IoU of 0.44, double the baseline on ORE_Random_50, and 10% better than the baseline on ORE_Random_25.

Additionally, while the performance of the fine-tuned models on the ORE_Random_15 and ORE_Random_25 datasets does not fluctuate significantly, there are some differences in the case of ORE_Random_50. Indeed, the checkpoint Erasing_30-40 performs best, followed by Erasing_10-40, Erasing_10-20, and with the Erasing_10-20 checkpoint performing worst of the fine-tuned models. These results suggest that including large erasing areas during fine-tuning particularly improves performance in both cases with small and large occlusions.



(a) ORE_Random *(b) ORE_Random_Inpainting* *(c) ORE_Random_Combined*

Figure 6.4: *Performance of the baseline and the proposed fine-tuned checkpoints on (a) ORE_Random, (b) ORE_Random_Inpainting and (c) ORE_Random_Combined. Colour indicates the erasing area percentage.*

Performance on ShapeNet-Chair and Pix3D dataset. Figure 6.5 shows that the fine-tuned models perform in a fairly similar manner when evaluated on the ShapeNet dataset. The baseline performs best with a mean IoU of 0.56 compared to the mean IoU of 0.55 of the fine-tuned checkpoints. These results indicate that the fine-tuned models performed better on the ORE_Random datasets without overfitting on the ShapeNet dataset.

Similarly to the results on the ORE_Random_50, the fine-tuned models perform better than the baseline when evaluated on the Pix3D dataset (see Figure 6.5). Additionally, the order in the performance is the same, with Erasing_30-40 performing the best and Erasing_10-20 performing the worst of the fine-tuned models. Erasing_30-40 achieves a mean IoU of 0.21 that compared to the baseline value of 0.18 is an improvement of the 16%. This suggests that applying erasing, especially with large erasing area percentages, improves the generalisation of the model since they all perform better than the baseline on Pix3D, which is a completely different dataset than ShapeNet composed of real-world images.

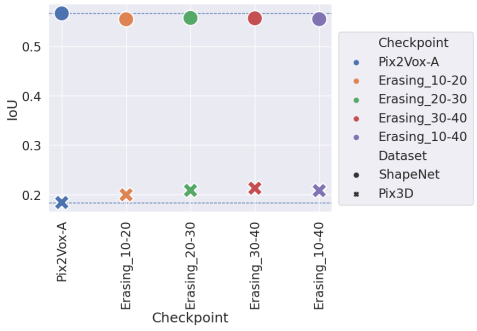


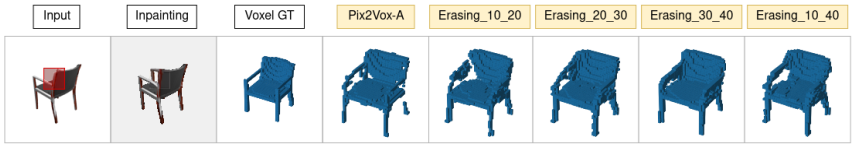
Figure 6.5: *Performance of the baseline and the proposed fine-tuned checkpoints on the ShapeNet-Chair (Chang et al. 2015) and Pix3D (Sun et al. 2018) datasets.*

6.2.2 3D reconstruction after applying inpainting

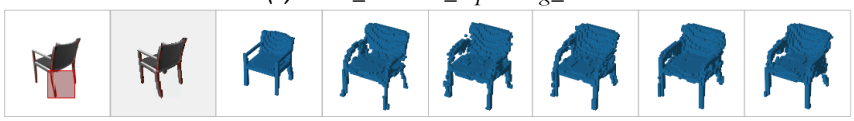
In this section, we evaluate the fine-tuned models on the ORE_Random_Inapainting datasets. Qualitatively, the inpainting results are very similar to the ground truth images, independently of the erasing area size, as can be seen in Figure A.13. Such level of results were achieved without fine-tuning the StandardDiffusion (Rombach et al. 2021) model on the ShapeNet dataset. However, there are some unwanted artefacts or wrong reconstructions. In particular, it can be seen the border of the erasing area on the inpainted images. Even after experimenting with different hyperparameters, such as *mask blur*, which allows to blur the border of the mask, the erasing border continued to be consistently evident.

Performance on ORE_Random_Inpainting datasets. After applying inpainting, the performance gap performance between the baseline and the fine-tuned models lowers. However, this is mostly due to the inferior performance of the fine-tuned models (see Figure 6.4b). In addition, the performance gap between different datasets also decreased, even though there still is a performance decreases with the increase of the erasing area. Indeed, the average difference of mean IoU values between the ORE_Random_Inpainting_25 and ORE_Random_Inpainting_50 is 14%, which is 7% less compared to the ORE_Random case and also includes the performance of the baseline. Interestingly, the Erasing_30-40 checkpoint still performs best overall cases, even though the baseline closely follows its performance; with a mean IoU that differs by 0.01 in the first two cases and 0.02 in the last case. Finally, Erasing_10-20 demonstrates the lowest performance.

Performance on ORE_Random_Combined datasets. Figure 6.4c shows the results of the last proposed approach. After combining both the occluded and the inpainted images, the fine-tuned



(a) ORE_Random_Inpainting_15



(b) ORE_Random_Inpainting_25

(c) ORE_Random_Inpainting_50

Figure 6.6: *3D voxel reconstruction of the proposed fine-tuned models and the baseline on sample 3a0e392db610f1a1504d5af97121b5f of the (a) ORE_Random_Inpainting_15, (b) ORE_Random_Inpainting_25 and (c) ORE_Random_Inpainting_50 datasets.*

models slightly improved their performance over the baseline. The fine-tuned results achieve similar results for the first two datasets while differing in the case of the last dataset where Erasing_30-40 performs best.

We experimented with switching the order of the inputs: feeding first the inpainted image and then the occluded image, but the results are identical, confirming that the context-aware module is not influenced by the input order.

Comparing all approaches. Finally, we compared all the previously seen performances across the proposed approaches, considering only the baseline and Erasing_30-40, the top-performing fine-tuning model. As depicted in Figure 6.7, utilising Erasing_30-40 without applying inpainting is the most effective approach overall. This method displays an average performance of 18% superior to the inpainting approach and a 4% advantage over the combined approach.

On the other hand, the optimal approach to use with the baseline varies based on the dataset. In the case of the ORE_Random_15 dataset, the results of the combined approach and the approach without inpainting are the same: both 0.47 mean IoU. Applying inpainting instead leads to a decrease in the performance of 14%. These results suggest that the Pix2Vox Contex-aware module entirely prefers the coarse volume generated from the occluded images, ignoring the volume generated from the inpainted images.

When applied to the ORE_Random_25 dataset, using the baseline model with the combined approach is preferable. The gain in performance is 7% compared to not without inpainting and 5% compared to the method of applying inpainting. Hence, the Contex-aware module combines the two coarse volumes generated from both the occluded and inpainted images.

Finally, in the case of the ORE_Random_50 dataset, using the baseline after applying inpainting leads to a gain of 51% in performance compared to avoiding applying inpainting and a gain of 9% compared to the combined approach. This case is in contrast to the previous instances, since the hallucinated filling from the inpainted images leads, on average, to a better reconstruction.

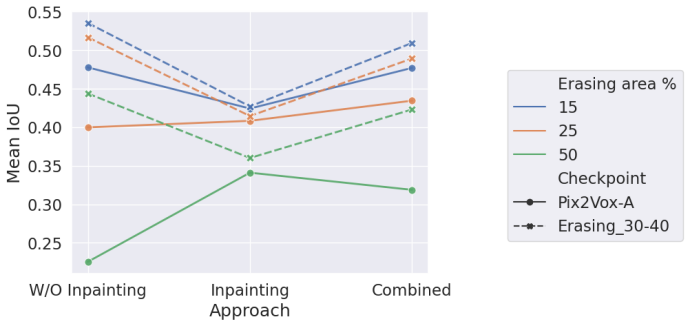


Figure 6.7: *Performance of the baseline and the proposed fine-tuned checkpoints on the ORE_Random_Combined datasets.*

6.2.3 Current limitations and future work

MORE at random locations. Using the MORE scheme lead to an information loss closer to the target value than the use of ORE and IRE schemes, when applied at centre. Only ORE applied at random achieved a better result. Hence, it could be explored if applying MORE at random, instead of at the centre, further improves the accuracy of the information loss on the ShapeNet dataset.

Improvement of the inpainting results. Applying inpainting greatly improves the performance of the baseline model only in severe cases of occlusion (50%). To slightly improve or match the performance of the baseline model on lighter occlusions, the inpainted images need to be combined with the partially occluded images. Additionally, fine-tuning the model is a more effective approach.

Perhaps, this is due to the inpainting results not being qualitatively good enough, generating new information that is not plausible. Another option could be that the generated filling for the cases of 15% and 25% occlusion could have the effect of an adversarial attack (Goodfellow et al. 2015). Adversarial attacks are changes to the input images that are imperceptible to the human eye, but cause the model to incorrectly classify an input that was previously classified correctly. In this case, the inpainted image could be qualitatively acceptable to a human observer, but the added information leads to a worse 3D reconstruction.

Missing segmentation step. While the aim is to be able to use 3D reconstruction in scenarios and settings that are not controlled or fixed, the proposed approach is effective if receives as inputs only images paired with the objects' segmentation masks. Indeed, the pipeline does not include a segmentation step. This means that images of occluded objects without a segmentation mask cannot be inpainted and hence are treated as Pix2Vox would normally do.

Generation of a realistic dataset. Experiments on using StableDiffusion (Rombach et al. 2021) for the generation of a realistic version of the ShapeNet dataset arose various issues that made the process impracticable. One major issue is the inconsistency with the newly generated pose of the object, compared to the original, and the poor results in generating images of the top and back view. However, *ControlNet* (Zhang and Agrawala 2023) could be used to solve this issue. In general, it allows fine-tuning latent-diffusion models, such as StableDiffusion, to learn a task-specific input condition. Applied to this case, ControlNet can be used to fine-tune StableDiffusion to consistently preserve the pose or correctly generate images of a new pose given the target pose and an image of the object.

7 Conclusion

This project has explored the field of 3D reconstruction and identified the gap in research in the case of object occlusion and the limitations of the currently available datasets used for training and evaluation. It is proposed a collection of occluded versions of the Chair class from the ShapeNet dataset (Chang et al. 2015): the IRE datasets, the ORE datasets and the MORE dataset. Such datasets have been generated by applying a combination of erasing augmentation methods and a novel scheme, MORE, for applying erasing. The generated dataset can be further divided based on the erasing area percentage removed, which is 15%, 25% and 50%. Additionally, the challenges of generating a realistic version of Shpaenet, using StableDiffusion (Rombach et al. 2021), have been highlighted.

The solutions proposed for improving single-image 3D reconstruction under objects inter-object occlusion included fine-tuning of Pix2Vox and inpainting using StableDiffusion. The proposed model, Erasing_30-40, is the result of fine-tuning the Pix2Vox-A model using a modified version of the suggested occluded synthetic images. Erasing_30-40 improves the performance of the baseline, Pix2Vox-A, of 97% in case of severe occlusion (50%). Additionally, it achieves 12% and 29% better results, compared to the baseline, in cases of 15% and 25% occlusion respectively. Finally, Erasing_30-40 is able to better generalise to real-world images, improving the results of 16% compared to Pix2Vox-A. Using the baseline after applying inpainting improves its performance by 51% in case of severe occlusion.

Future work directions include the use of the recently proposed *ControlNet* (Zhang and Agrawala 2023) to overcome the main limitations of using StableDiffusion for generating a realistic version of the ShapeNet dataset. The proposed approach can be improved by implementing a segmentation step for autonomously detecting the inpainting area. Additionally, it could be further explored the impact of using inpainted images as input for 3D reconstruction.

A Appendices

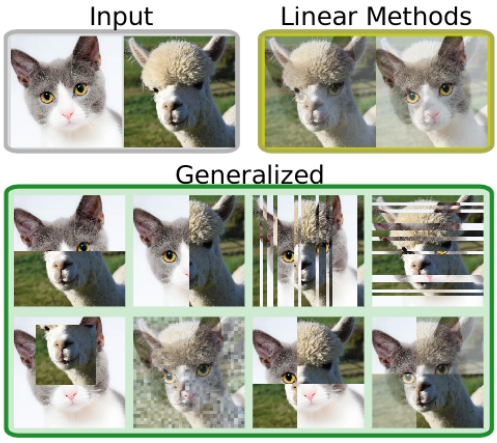


Figure A.1: *Linear and non-linear methods applied to two input images (Summers and Dinneen 2019)*

(a) *Original image* ***(b)*** *General samples* ***(c)*** *Back view samples*

Figure A.2: *(a) shows the ShapeNet chair model id 1aa07508b731af79814e2be0234da26c used to fine-tune StableDiffusion (SD) (Rombach et al. 2021) using Dreambooth (Ruiz et al. 2022). (b) shows images of the 1aa07508b731af79814e2be0234da26c samples generated with Dreambooth and SD, while (c) shows samples generated with the request of depicting the back view of sample 1aa07508b731af79814e2be0234da26c.*

Figure A.3: *Different inpainting results generated giving the input showed in Figure 5.3 and using different CFG values (CFG=1, CFG=7 and CFG=20).*

Figure A.4: *Samples of inpainting results generated giving the input showed in Figure 5.3 and selecting different inpainting areas.*

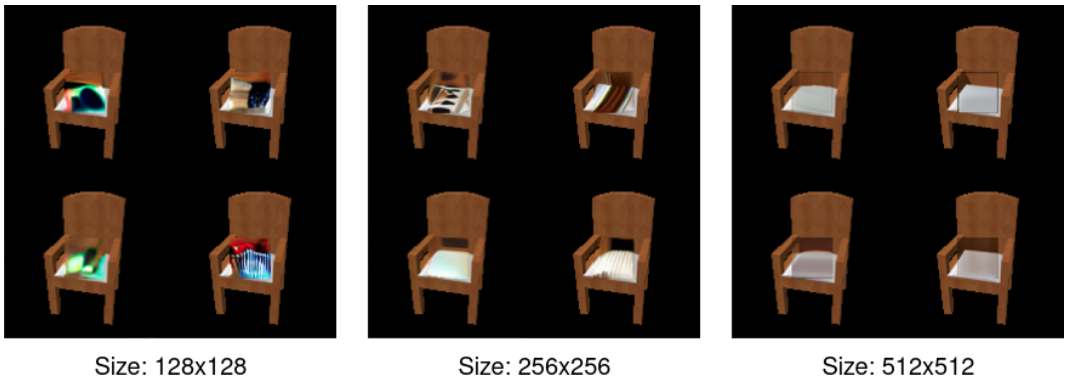


Figure A.5: *Various inpainting results using the input showed in Figure 5.3 and different resolution sizes (128x128, 256x256 and 512x512).*

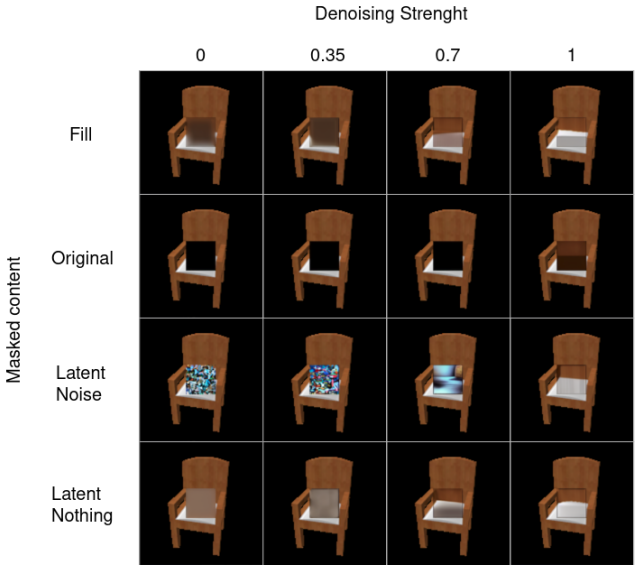
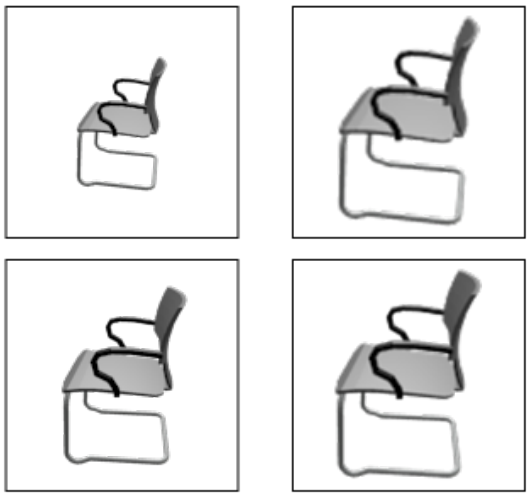
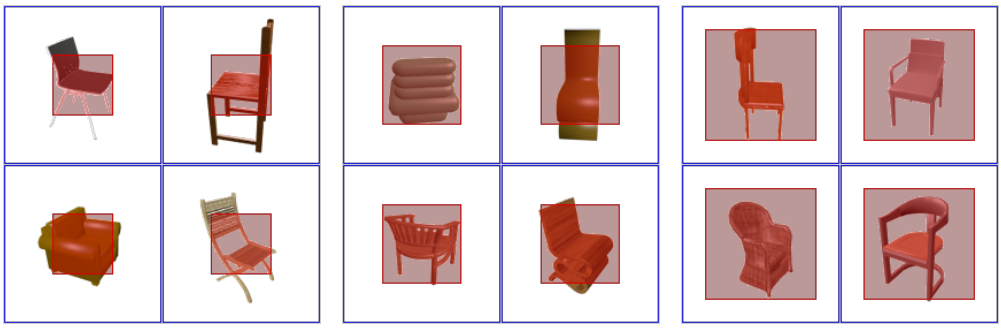


Figure A.6: *Different inpainting results generated giving the input showed in Figure 5.3 and using different denoising strength and inpaint fill values.*



(a) *Original ShapeNet* ***(b)*** *Cropping and resizing*

Figure A.7: *(a) shows samples no.07 and no.17, respectively, of ShapeNet chair model id 1a74a83fa6d24b3cacd67ce2c72c02e. (b) Shows the results after applying cropping and resizing on such images.*

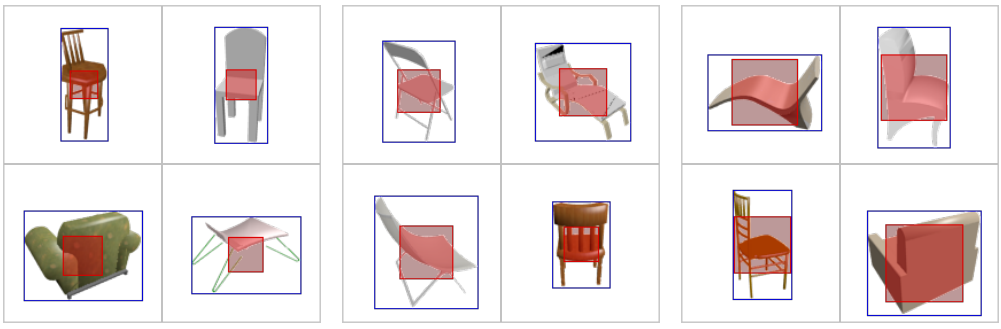


(a) IRE_Center_15 *(b)* IRE_Center_25 *(c)* IRE_Center_50

Figure A.8: *Samples of the IRE datasets with mask applied at the centre of the image and different erasing area percentage*

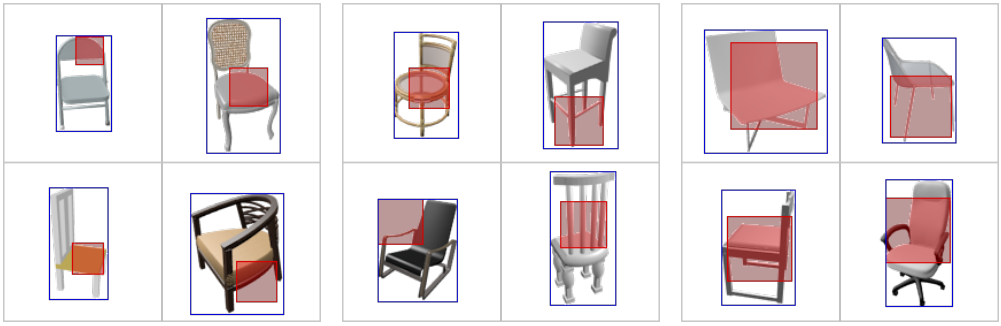
(a) IRE_Random_15 *(b)* IRE_Random_25 *(c)* IRE_Random_50

Figure A.9: *Samples of the IRE datasets with mask applied at random locations and different erasing area percentage*



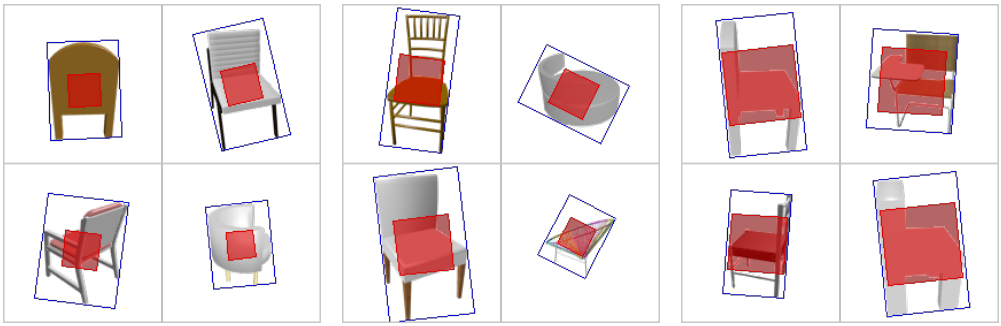
(a) ORE_Center_15 *(b)* ORE_Center_25 *(c)* ORE_Center_50

Figure A.10: *Samples of the ORE datasets with mask applied at the centre of the object bounding box and different erasing area percentage*



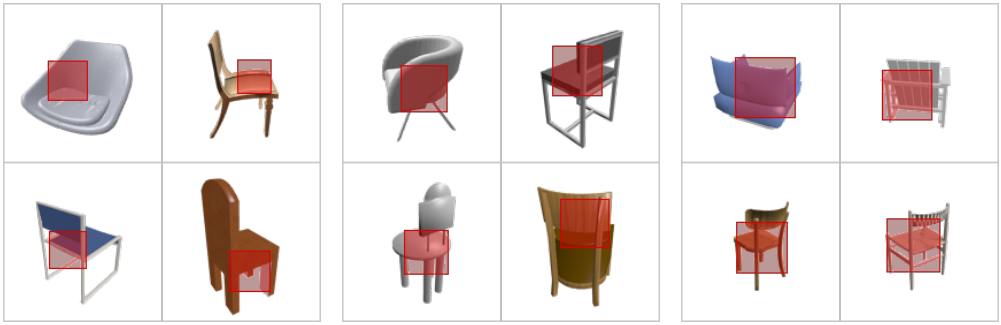
(a) ORE_Random_15 *(b)* ORE_Random_25 *(c)* ORE_Random_50

Figure A.11: *Samples of the ORE datasets with the erasing applied at random locations, within the object bounding box) and different erasing area percentage*



(a) MORE_Center_15 *(b)* MORE_Center_25 *(c)* MORE_Center_50

Figure A.12: *Samples of the MORE datasets with erasing applied at the centre of the object minimum bounding box and different erasing area percentage*



(a) ORE_Random_15 *(b)* ORE_Random_25 *(c)* ORE_Random_50

(d) ORE_Random_Inpainting_15 *(e)* ORE_Random_Inpainting_25 *(f)* ORE_Random_Inpainting_50

Figure A.13: *Samples of the ORE_Random datasets before (a) (b) (c) and after (d) (e) (f) having applied inpainting.*

7 Bibliography

W. Baatz, M. Fornasier, P. A. Markowich, , and C.-B. Schönlieb. Inpainting of ancient austrian frescoes. *CorpusID:11414112*, 2008.

Y. Ben-Shabat, X. Yu, F. S. Saleh, D. Campbell, C. Rodriguez-Opazo, H. Li, and S. Gould. The ikea asm dataset: Understanding people assembling furniture through actions, objects and pose. *arxiv:2007.00394*, 7 2020.

A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, and et al. Shapenet: An information-rich 3d model repository. *arxiv:1512.03012*, Dec 2015.

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. *arXiv:1604.00449*, 2016.

M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. *arXiv:1311.3618*, 2013.

M. Dahnert, J. Hou, M. Nießner, and A. Dai. Panoptic 3d scene reconstruction from a single rgb image. *arXiv:2111.02444*, 2021.

T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arxiv.org:1708.04552*, 8 2017a.

T. DeVries and G. W. Taylor. Dataset augmentation in feature space. *arxiv.org:1702.05538*, 2017b.

P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *arxiv:2105.05233*, 2021.

C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. A. Efros. What makes paris look like paris? *ACM Transactions on Graphics (SIGGRAPH)*, 31(4):101:1–101:9, 2012.

R. Dovgard and R. Basri. Statistical symmetric shape from shading for 3d structure recovery of faces. In T. Pajdla and J. Matas, editors, *Computer Vision - ECCV 2004*, pages 99–113, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-24671-8.

K. Fu, J. Peng, Q. He, and H. Zhang. Single image 3d object reconstruction based on deep learning: A review. *Multimedia Tools and Applications*, 80:1–36, 01 2021. doi: 10.1007/s11042-020-09722-8.

L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arxiv.org:1508.06576*, 2015.

R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. *arxiv:arXiv:1603.08637*, 2016.

R. Girshick. Fast r-cnn. *arXiv:1504.08083*, 2015.

G. Gkioxari, J. Malik, and J. Johnson. Mesh r-cnn. *arXiv:1906.02739*, 2019.

I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, Y. Zhou, C. Ramaiah, F. Feng, R. Li, X. Wang, D. Athanasakis, J. Shawe-Taylor, M. Milakov, J. Park, R. Ionescu, M. Popescu, C. Grozea, J. Bergstra, J. Xie, L. Romaszko, B. Xu, Z. Chuang, and Y. Bengio. Challenges in representation learning: A report on three machine learning contests. In M. Lee, A. Hirose, Z.-G. Hou, and R. M. Kil, editors, *Neural Information Processing*, pages 117–124, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-42051-1.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv:1406.2661*, 2014.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arxiv.org:1412.6572*, 2015.

T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation. *arXiv:1802.05384*, 2018.

X.-F. Han, H. Laga, and M. Bennamoun. Image-based 3d object reconstruction: State-of-the-art and trends in the deep learning era. *arxiv.org:1708.04552*, 6 2019. doi: 10.1109/TPAMI.2019.2954885.

J. Hays and A. A. Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007.

J. Ho and T. Salimans. Classifier-free diffusion guidance. *arxiv.org:2207.12598*, 2022.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *arxiv.org:2006.11239*, 6 2020.

L. Hoang, S.-H. Lee, O.-H. Kwon, and K.-R. Kwon. A deep learning method for 3d object classification using the wave kernel signature and a center point of the 3d-triangle mesh. *Electronics*, 8(10), 2019. ISSN 2079-9292. doi: 10.3390/electronics8101196. URL https://www.mdpi.com/2079-9292/8/10/1196.

X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. *arxiv.org:1804.04732*, 2018.

C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. *arxiv:1704.00710*, 2017.

H. Inoue. Data augmentation by pairing samples for images classification. *arxiv.org:1801.02929*, 2018.

J. Jam, C. Kendrick, V. Drouard, K. Walker, G.-S. Hsu, and M. H. Yap. R-mnet: A perceptual adversarial network for image inpainting. *arXiv:2008.04621*, 2020.

J. Jam, C. Kendrick, K. Walker, V. Drouard, J. G.-S. Hsu, and M. H. Yap. A comprehensive review of past and present image inpainting methods. *Computer Vision and Image Understanding*, 203: 103147, 2021. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2020.103147. URL https://www.sciencedirect.com/science/article/pii/S1077314220301661.

J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *arxiv.org:1603.08155*, 2016.

T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv:1710.10196*, 2017.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv:1512.09300*, 2015.

B. Y. Lee, L. H. Liew, W. S. Cheah, and Y. C. Wang. Occlusion handling in videos object tracking: A survey. *IOP Conference Series: Earth and Environmental Science*, 18(1):012020, feb 2014. doi: 10.1088/1755-1315/18/1/012020. URL https://dx.doi.org/10.1088/1755-1315/18/1/012020.

D. Lewy and J. Mańdziuk. An overview of mixing augmentation methods and augmentation strategies. *Artificial Intelligence Review*, 2022. ISSN 15737462. doi: 10.1007/s10462-022-10227-z.

B. Li, F. Wu, S.-N. Lim, S. Belongie, and K. Q. Weinberger. On feature normalization and data augmentation. *arxiv.org:2002.11102*, 2021.

S. K. Lim, Y. Loo, N.-T. Tran, N.-M. Cheung, G. Roig, and Y. Elovici. Doping: Generative data augmentation for unsupervised anomaly detection with gan. *arxiv.org:1808.07632*, 2018.

T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context. *arxiv.org:1405.0312*, 2015.

G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro. Image inpainting for irregular holes using partial convolutions. *arXiv:1804.07723*, 2018.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. *arxiv:1411.7766*, 2014.

R. G. Lopes, D. Yin, B. Poole, J. Gilmer, and E. D. Cubuk. Improving robustness without sacrificing accuracy with patch gaussian augmentation. *arxiv.org:1906.02611*, 2019.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. V. Gool. Repaint: Inpainting using denoising diffusion probabilistic models. *arxiv:2201.09865*, 1 2022.

M. Michalkiewicz, J. K. Pontes, D. Jack, M. Baktashmotlagh, and A. Eriksson. Implicit surface representations as layers in neural networks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4742–4751, 2019. doi: 10.1109/ICCV.2019.00484.

A. Newson, A. Almansa, M. Fradet, Y. Gousseau, and P. Pérez. Video inpainting of complex scenes. *SIAM Journal on Imaging Sciences*, 7(4):1993–2019, jan 2014. doi: 10.1137/140954933. URL https://doi.org/10.1137%2F140954933.

S. Parida, V. Srinivas, B. Jain, R. Naik, and N. Rao. Survey on diverse image inpainting using diffusion models. In *2nd International Conference on Paradigm Shifts in Communications Embedded Systems, Machine Learning and Signal Processing (PCEMS)*, pages 1–5. Institute of Electrical and Electronics Engineers (IEEE), 6 2023. ISBN 9798350310719. doi: 10.1109/pcems58491.2023.10136091.

D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. *arxiv.org:1604.07379*, 2016.

Z. Qin, Q. Zeng, Y. Zong, and F. Xu. Image inpainting based on deep learning: A review. *Displays*, 69, 2021. doi: 10.1016/j.displa.2021.102028. URL https://doi.org/10.1016/j.displa.2021.102028.

F. Rengier, A. Mehndiratta, H. von Tengg-Kobligk, C. M. Zechmann, R. Unterhinninghofen, H.-U. Kauczor, and F. L. Giesel. 3d printing based on imaging data: review of medical applications. *International Journal of Computer Assisted Radiology and Surgery*, 5(4):335–341, may 2010. doi: 10.1007/s11548-010-0476-x. URL https://doi.org/10.1007%2Fs11548-010-0476-x.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. *arxiv:2112.10752*, 12 2021.

N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arxiv.org:2208.12242*, 2022.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *arXiv:1409.0575*, 2014.

C. Saharia, W. Chan, H. Chang, C. A. Lee, J. Ho, T. Salimans, D. J. Fleet, and M. Norouzi. Palette: Image-to-image diffusion models. *arXiv:2111.05826*, 2021.

C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6, 12 2019. ISSN 21961115. doi: 10.1186/s40537-019-0197-0.

A. Sinha, A. Unmesh, Q. Huang, and K. Ramani. Surfnet: Generating 3d shape surfaces using deep residual networks. *arXiv:1703.04079*, 2017.

E. Smith and D. Meger. Improved adversarial systems for 3d object generation and reconstruction. *arXiv:1604.00449*, 2017.

J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *arxiv:1503.03585*, 3 2015.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56): 1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

C. Summers and M. J. Dinneen. Improved mixed-example data augmentation. *arxiv.org:1805.11272*, 2019.

X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. *arxiv.org:1804.04610*, 2018.

M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. *arxiv:1703.09438*, 2017.

J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *arxiv.org:1703.06907*, 2017.

R. Tovey, M. Benning, C. Brune, M. J. Lagerwerf, S. M. Collins, R. K. Leary, P. A. Midgley, and C.-B. Schönlieb. Directional sinogram inpainting for limited angle tomography. *Inverse Problems*, 35(2):024004, jan 2019. doi: 10.1088/1361-6420/aaf2fe. URL https://doi.org/10.1088%2F1361-6420%2Faaf2fe.

S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. *arxiv:1704.00710*, 2017.

P. Vitoria and C. Ballester. Automatic flare spot artifact detection and removal in photographs. *Journal of Mathematical Imaging and Vision*, 61:515–533, 2019. doi: 10.1007/s10851-018-0859-0. URL https://doi.org/10.1007/s10851-018-0859-0.

P.-S. Wang. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *arxiv:1712.01537*, 2017.

J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *arXiv:1610.07584*, 2016.

J. Wu, Y. Wang, T. Xue, X. Sun, W. T. Freeman, and J. B. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5d sketches. *arXiv:1901.11153*, 2017.

Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. *arxiv:arXiv:1406.5670*, 2014.

H. Xiang, Q. Zou, M. A. Nawaz, X. Huang, F. Zhang, and H. Yu. Deep learning for image inpainting: A survey. *Pattern Recognition*, 134, 2 2023. ISSN 00313203. doi: 10.1016/j.patcog.2022.109046.

Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82, 2014. doi: 10.1109/WACV.2014.6836101.

H. Xie, H. Yao, X. Sun, S. Zhou, and S. Zhang. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. *arxiv.org:1901.11153*, 2019.

S. Yadav. Implicit vs parametric 3d shape representation, Oct 2022. URL https://medium.com/p/9d4c01c8c60c.

S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen. Image data augmentation for deep learning: A survey. *arxiv.org:2204.08610*, 2022a.

Z. Yang, Z. Zhang, and Q. Huang. Hm3d-abo: A photo-realistic dataset for object-centric multi-view 3d reconstruction. *arXiv:2206.12356*, 2022b.

D. Zeng, R. Veldhuis, and L. Spreeuwers. A survey of face recognition techniques under occlusion. *IET Biometrics*, 10(6):581–606, 2021. doi: https://doi.org/10.1049/bme2.12029. URL https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/bme2.12029.

L. Zhang and M. Agrawala. Adding conditional control to text-to-image diffusion models. *arxiv.org:2302.05543*, 2023.

X. Zhang, Z. Zhang, C. Zhang, J. B. Tenenbaum, W. T. Freeman, and J. Wu. Learning to reconstruct shapes from unseen classes. *arXiv:1812.11166*, 2018.

Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. *arxiv.org:1708.04896*, 8 2017.

B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva. Places: An image database for deep scene understanding. *arXiv:1610.02055*, 2016.

J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017a. doi: 10.1109/ICCV.2017.244.

X. Zhu, Y. Liu, Z. Qin, and J. Li. Data augmentation in emotion classification using generative adversarial networks. *arxiv.org:1711.00648*, 2017b.