

Generating Synthetic X-rays Using Generative Adversarial Networks

Mustafa Haiderbhai

Thesis submitted to the University of Ottawa
In partial fulfillment of the requirements for the
M.A.Sc. degree in
Biomedical Engineering

Ottawa-Carleton Institute of Biomedical Engineering (OCIBME)
Faculty of Engineering
University of Ottawa

© Mustafa Haiderbhai, Ottawa, Canada, 2020

Abstract

We propose a novel method for generating synthetic X-rays from atypical inputs. This method creates approximate X-rays for use in non-diagnostic visualization problems where only generic cameras and sensors are available. Traditional methods are restricted to 3-D inputs such as meshes or Computed Tomography (CT) scans. We create custom synthetic X-ray datasets using a custom generator capable of creating RGB images, point cloud images, and 2-D pose images. We create a dataset using natural hand poses and train general-purpose Conditional Generative Adversarial Networks (CGANs) as well as our own novel network *pix2xray*. Our results show the successful plausibility of generating X-rays from point cloud and RGB images. We also demonstrate the superiority of our *pix2xray* approach, especially in the troublesome cases of occlusion due to overlapping or rotated anatomy. Overall, our work establishes a baseline that synthetic X-rays can be simulated using inputs such as RGB images and point cloud.

Acknowledgments

I would like to acknowledge and thank my supervisor Dr. Pascal Fallavollita for supervising this thesis work, and for his outstanding support and guidance through the years. I would also like to thank Dr. Sergio Ledesma, Dr. Nassir Navab, Dr. Phillipp Furnstahl, Matthias Seibold, Sing Chun Lee, who all provided guidance and support throughout this research work.

I am grateful to my parents, siblings, and to my lovely fiancé, for their never-ending support and encouragement, without which I would not have made it this far in my research career.

I would like to acknowledge and thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for providing financial assistance to this project through funding and scholarships.

Table of Contents

List of Tables	vi
List of Figures	vii
Abbreviations	x
1 Introduction	1
1.1 Synthetic X-rays	1
1.2 Generative Adversarial Networks (GANs)	5
1.3 Motivation	8
1.4 Objectives	10
1.5 Technical Contributions	10
1.6 Thesis Steps	11
1.7 Thesis Outline	11
2 State of the Art	12
2.1 Digitally Reconstructed Radiography	12
2.2 Data Augmentation	14
2.3 X-ray Synthesis From Text	15
2.4 Medical Image Translation	17
3 Methodology	19
3.1 Synthetic Data Generator	19

3.2	Point Cloud Images	22
3.3	RGB Images and Pose to Resolve Occlusion	24
3.4	3-D Point Cloud Data	29
3.5	Validation	31
4	Results and Discussion	33
4.1	Point Cloud Images	33
4.2	RGB Images and Pose	40
4.3	3D Point Cloud Data	48
5	Conclusion	53
5.1	Summary	53
5.2	Objectives	54
5.3	Future Work	55
5.3.1	Data Variation	55
5.3.2	Anatomy	55
5.3.3	Partial Point Clouds	56
5.3.4	User Validation	56
	References	57

List of Tables

4.1	Evaluation of image similarity between the predicted images of different point cloud density inputs with their respective networks and the ground truth.	34
4.2	Summary of qualitative differences between point cloud density predicted X-rays.	34
4.3	Evaluation metrics of image similarity between the two general-purpose architectures <i>CycleGAN</i> and <i>pix2pix</i> , and our own approach <i>pix2xray</i>	41
4.4	Evaluation metrics of image similarity in unique test dataset with only occluded cases.	41
4.5	Summary of qualitative differences between outputs of networks using RGB input.	41
4.6	Evaluation metrics of image similarity comparing <i>pix2pix</i> and our own <i>pix2xray</i> to their modified counterparts that include 3-D point cloud processing.	48
4.7	Evaluation metrics of image similarity in unique test dataset with only occluded cases between the 3-D point cloud networks and the originals.	48
4.8	Summary of qualitative differences between outputs of networks using RGB and point cloud input together.	49
4.9	Comparison of network complexity between networks based on number of parameters and time taken to test 10 images in seconds.	52

List of Figures

1.1	a,b) Ground truth images and c,d) Monte Carlo simulated images compared using voxel-based Monte Carlo Simulation [1].	3
1.2	Process of generating a synthetic X-ray using ray-casting used in <i>gVirtualXray</i> [2].	4
1.3	Sample of synthetic X-rays generated using <i>gVirtualXray</i> from Vidal et al. [2].	5
1.4	Realistic faces randomly generated by the Progressive GAN architecture trained on the CELEBA-HQ dataset [3].	6
1.5	Example results of generating synthetic images based on text input done by Reed et al. [4].	6
1.6	Architecture of a typical GAN network	7
1.7	Top-left shows a CAM-C system using a video camera, which produces RGB images that can be used for an AR overlay as shown on the bottom. The top-right shows a CAM-C system with depth cameras. Synthetic X-rays can be generated from RGB images taken from video or depth sensors, and from 2-D/3-D point clouds taken from depth sensors [5].	9
2.1	GPU-based DRR method created by Dorgham et al., showing a) typical ray-casting method b) DRR raycasting method c) render of the model used and d) sample DRR generated using their algorithm [6].	13
2.2	Results of applying Progressive GAN towards mammography images done by Korkinof et al., images of higher resolution produced as training progresses [7].	15

2.3	Text to X-ray image synthesis conducted by Yang et al., examining popular high-resolution GAN architectures and their novel <i>XRayGAN</i> architecture, where X-ray images are generated using text descriptions [8]. Excerpts from the text input for the shown output samples include: a) increased opacity in upper right lobe (orange) and left mid-lung (purple) b) normal lungs with slight degeneration in thoracic spine c) Opacity artifact over right apex and mediastinum and coarse interstitial markings.	16
2.4	Generating synthetic images from surface geometry accomplished by Teixeira et al., a) surface geometry b) ground truth c) generated image and d) generated body markers to allow for data augmentation	18
3.1	Hand model as seen in Blender, with the inner and outer meshes visible as well as the armature which is used to make poses.	20
3.2	Screen-shot of our synthetic data generator application, which hosts a real-time X-ray simulation and the ability to randomly create large datasets. . .	21
3.3	Overall work-flow for training the point cloud to X-ray translation networks starting with a) sampling 10 natural hand poses from the NYU hand pose dataset b) creating 10 meshes with the generated poses c) generating the six synthetic datasets of 1000 image pairs through randomization of each pose 100 times in the generator and d) training the six networks.	23
3.4	Overview of the pix2pix GAN architecture.	24
3.5	Overall work-flow for training the RGB image to X-ray networks starting with a) sampling 5 natural hand poses from the NYU dataset b) generating 5 hand meshes c) importing into generator to create image triplets d) generating a dataset 1000 image triplets through randomization of each pose 200 times and e) training three networks: <i>pix2pix</i> , <i>CycleGAN</i> , and <i>pix2xray</i> . . .	25
3.6	Sample of image triplets created through the generator, each triplet consisting of a RGB image, pose image, and X-ray image.	26
3.7	Overview of our modified architecture <i>pix2xray</i> in comparison with the original <i>pix2pix</i> architecture.	27
3.8	Overview of CycleGAN architecture.	29
3.9	Architecture overview of our modified <i>pix2xray</i> including the <i>PointNet</i> branch for point cloud processing.	31

4.1	Sample of predicted X-rays generated using the 50K point cloud density network and dataset.	35
4.2	Sample of predicted X-rays from each of the six different networks using a similar pose. The first column of each pair shows the ground truth X-ray, the second column shows the predicted X-ray generated from a point cloud image input of the density specified on the left.	36
4.3	Sample of low 5K (top) and high 50K (bottom) point cloud density input that produces a visible failure case.	37
4.4	Enlargement of 5K and 10K predicted X-rays to visualize the inconsistencies in these low density inputs.	38
4.5	Enlargement of 50K and 20K predicted X-rays to visualize the inconsistencies and difference between high density inputs.	39
4.6	Comparison of predicted X-rays between <i>CycleGAN</i> , <i>pix2pix</i> , and our network <i>pix2xray</i> using sample inputs.	42
4.7	Comparison of predicted X-rays when handling occlusion between <i>CycleGAN</i> , <i>pix2pix</i> , and our network <i>pix2xray</i> using occluded sample inputs.	43
4.8	Comparison of predicted X-rays to real X-rays taken from a bone phantom, using similar poses generated in our generator.	44
4.9	Enlargement of predicted X-rays generated from <i>CycleGAN</i> , <i>pix2pix</i> , and our network <i>pix2xray</i> , to visualize the inconsistencies.	45
4.10	Enlargement of predicted X-rays generated from <i>CycleGAN</i> , <i>pix2pix</i> , and our network <i>pix2xray</i> , from the occluded test set to visualize the inconsistencies.	45
4.11	Comparison of predicted X-rays between <i>pix2pix</i> , <i>pix2xray</i> , and their modified counterparts that include <i>PointNet</i> modifications for 3-D point cloud input handling.	49
4.12	Comparison of predicted X-rays when handling occlusion between <i>pix2pix</i> , <i>pix2xray</i> , and their modified <i>PointNet</i> counterparts using occluded samples as input.	50
4.13	Enlargement of predicted X-rays generated from modified networks to visualize the inconsistencies.	51
4.14	Enlargement of predicted X-rays generated from modified networks using samples from our occluded set to visualize the inconsistencies.	52

Abbreviations

AR Augmented Reality [vii](#), [8](#), [9](#), [22](#), [52](#), [56](#)

CAM-C Camera Augmented Mobile C-arm [vii](#), [8](#), [9](#), [22](#), [55](#)

CGAN Conditional Generative Adversarial Network [5](#), [7](#), [8](#), [10](#), [11](#), [22](#), [40](#)

CPU Central Processing Unit [2](#)

CT Computed Tomography [1](#), [8](#), [12](#), [14](#), [53](#)

DRR Digitally Reconstructed Radiograph [vii](#), [11–14](#)

GAN Generative Adversarial Network [vii](#), [viii](#), [5–8](#), [10](#), [12](#), [14–17](#), [35](#), [46](#), [47](#), [53](#)

GPU Graphics Processing Unit [vii](#), [2](#), [12–14](#), [20](#), [31](#), [52](#)

NYU New York University [viii](#), [19](#), [23](#)

PSNR Peak Signal to Noise Ratio [32–34](#), [41](#), [43](#), [48](#), [50](#)

RGB Red Green Blue [v–viii](#), [9–12](#), [24](#), [26](#), [27](#), [30](#), [40](#), [41](#), [48](#), [49](#), [53](#), [54](#), [56](#)

RGB-D Red Green Blue and Depth [40](#)

RMSE Root Mean Squared Error [32–34](#), [41](#), [43](#), [48](#), [50](#)

SSIM Structural Similarity Index Metric [31](#), [33](#), [34](#), [41](#), [43](#), [48](#), [50](#)

Chapter 1

Introduction

1.1 Synthetic X-rays

X-ray imaging has existed since its inception in 1895 by Wilhelm Röntgen, and has become a staple technology in medical diagnostics [9]. Projectional radiography - the name for conventional X-ray radiography - has found prominent diagnostic value in imaging of the skeletal system, as well as soft tissue areas such as the lung, abdominal cavity, and cardiovascular system. Advances in X-ray technology has brought forward innovative applications and techniques, elevating pre-operative and intra-operative imaging options. [Computed Tomography \(CT\)](#) imaging is a process for generating three-dimensional images using multiple two-dimensional tomographic slices. Fluoroscopy utilizes real-time X-rays with fluoroscopic contrast agents to aid surgeons in image-guided procedures or to examine the movement of fluoroscopic objects to diagnose various disorders. These prominent and critical applications of X-rays in medicine make the study and advancement of X-ray technology essential.

X-ray devices are considered hazardous due to the ionizing radiation that individuals are exposed to when close to an active X-ray device [10]. Healthcare practitioners such as radiologists, cardiologists, and orthopedic surgeons are known to be at risk because of repeated long term usage from X-ray devices. This exposure is a necessary risk in procedures where X-ray imaging is crucial to success and efficacy, but numerous safety measures are used to decrease the dose of ionizing radiation [11]. Side effects of exposure to radiation have been well documented and can include skin erythema, necrosis, cataracts, and malignancies [12]. Outside of the safe dosage range used in healthcare devices, high doses of X-ray radiation can cause immediate burns and hair loss on exposed skin [13]. Therefore,

scientists and engineers will often use simulation technology to safely experiment, analyze data, and conceptualize possible outcomes using computerized models [14]. In this way, concepts, prototypes, and hypotheses can be iterated in a safe method before live testing.

Simulations use computerized three-dimensional models to perform calculations that can estimate the X-ray photon interaction with matter and therefore estimate the resulting image found on the detector. The Monte Carlo simulation method follows photons emitted by a point source within a given boundary until they interact with matter; events such as direction, interaction type, and energy are randomly sampled from predefined probability distributions [15, 1]. Monte Carlo simulations produce the most accurate simulations of X-ray imaging due to accurate physics modeling as well as tracking of secondary particles [16]. Figure 1.1 from Bottigli et al., shows the high image similarity produced using a Monte Carlo simulation, down to the complex microstructures [1]. Although analytical solutions could be used for simpler scenarios, the Monte Carlo method excels with complex geometry and experiments with higher-order interactions. The downside of using Monte Carlo algorithms is that they are computationally expensive, and therefore restrict the usage of X-ray simulations in modern applications where speed is required.

A faster alternative to the Monte Carlo method is the use of ray-tracing. In this method, X-rays are represented as geometric vectors found in a three-dimensional space that holds the object under study and a grid of pixels representing the detector. The rays are traced from the source through the object until it reaches the detector. The intersections of the ray with the model before reaching the detector are used to evaluate the attenuation and value of the pixel on the detector, thus creating an X-ray image [17]. Ray tracing algorithms solve the obstacle of computation time by simplifying the simulation while still maintaining a high level of accuracy. Ray-tracing can make the computation almost 100 times faster, bringing down processing time to under one second when dealing with 1000 by 1000 resolution image.

In recent years, advances in [Graphics Processing Unit \(GPU\)](#) technology has allowed for an increase in the speed of parallel computations, allowing for real-time simulation applications to become possible. GPU-accelerated computations can be used to create interactive visualizations, and simulations embedded into other software rather than stand-alone. *gVirtualXray* is an open-source software developed by Vidal et al. that uses the GPU to provide accelerated simulations of X-rays using the ray-tracing method [2]. Their work provides a method that is sixty times faster than [Central Processing Unit \(CPU\)](#)-based state of the arts, opening up the possibility for instantaneous real-time simulations. The result is a processing time that can be accomplished in milliseconds. As shown in

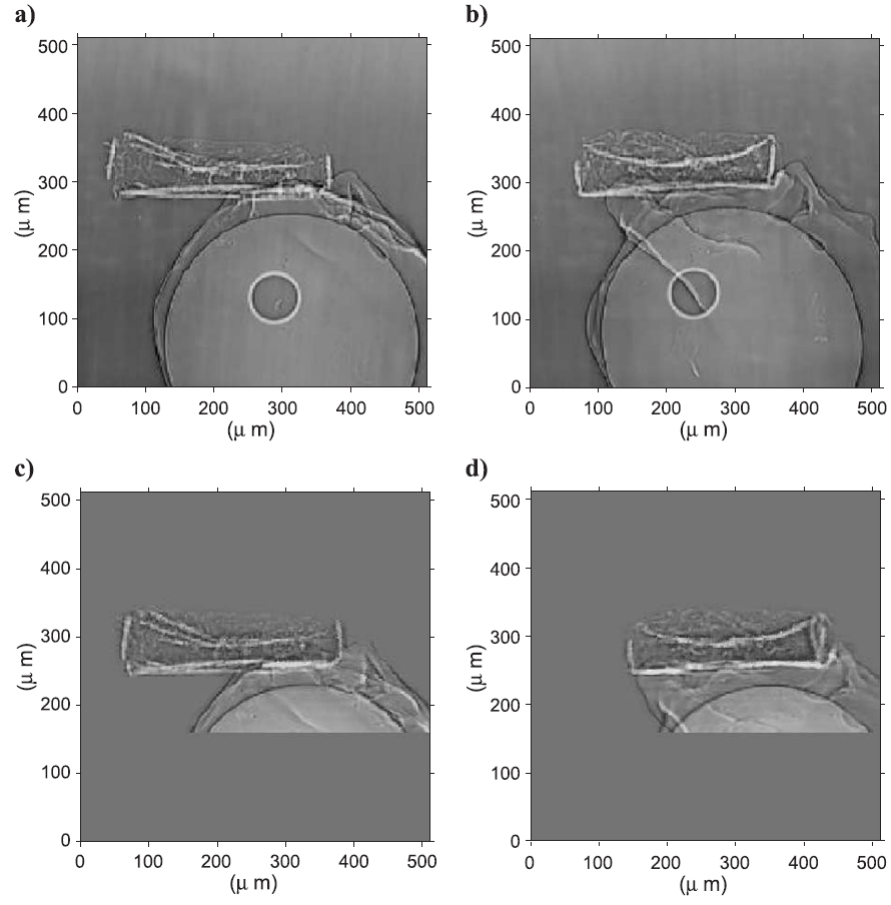


Figure 1.1: a,b) Ground truth images and c,d) Monte Carlo simulated images compared using voxel-based Monte Carlo Simulation [1].

Figure 1.2, casting a ray through model geometry produces the X-ray, and further medical examples produced by their library can be seen in Figure 1.3 The benefit of using this simpler but substantially faster method is shown in later works, where they achieve simulation of respiratory motion in X-ray [18]. This technology allows for more complete X-ray simulations, where the simulated X-ray is accompanied by rigid body physics and soft tissue deformation to allow for an immersive and accurate experience.

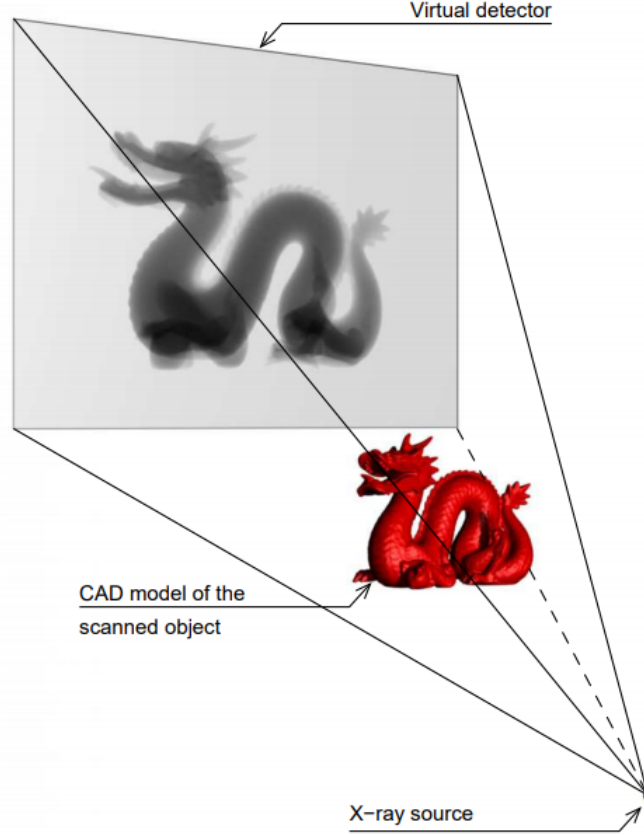


Figure 1.2: Process of generating a synthetic X-ray using ray-casting used in *gVirtualXray* [2].

The limitation of the above-mentioned approaches is the requirement of an accurate three-dimensional computerized model of the object under study to produce accurate results. This limitation restricts the possible applications of X-ray simulations to situations where such models are available, which further limits the generalizability of such technologies.

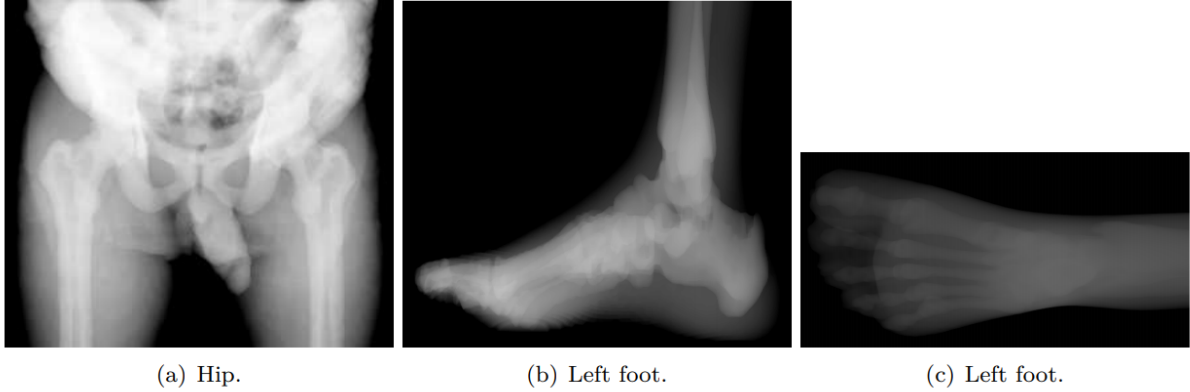


Figure 1.3: Sample of synthetic X-rays generated using *gVirtualXray* from Vidal et al. [2].

1.2 Generative Adversarial Networks (GANs)

Image-to-image translation is the concept of transforming an input image into a different output image and has been an area of constant study over the past few decades. An example of this problem can be seen in the extensively studied concept of image de-blurring, also known as image deconvolution [19]. This type of problem has classically been solved algorithmically, such as in the case of deconvolution where a cost function is minimized to determine the values of a blur kernel which can be applied to reduce the blurring of the image [20]. Another example of image translation is the concept of stylization, where images are transformed to radically change the appearance of the image content itself. A famous problem in this area is the transformation of the image from day to night or vice versa, referred to in some papers as *time hallucinations* [21]. The problem can again be solved algorithmically using a combination of scene matching followed by local affine transformations to create highly plausible results [21].

This area of study has recently seen a shift due to the rise and popularity of machine learning based approaches. The popularity of using machine learning is not without merit, specific network architectures and novel approaches have provided remarkable results that extend beyond the capabilities of traditional algorithmic approaches. One currently prominent new learning-based approach to image generation problems has been the use of **GAN**. Figure 1.4 showcases the results of Karras et al. and their Progressive **GAN** architecture, where a unique **GAN** architecture trained on celebrity faces can generate highly realistic new faces of humans that do not exist [3]. Figure 1.5 highlights the work of Reed et al., in generating images using only text input through a **Conditional Generative Adversarial Network (CGAN)** [4].



Figure 1.4: Realistic faces randomly generated by the Progressive GAN architecture trained on the CELEBA-HQ dataset [3].

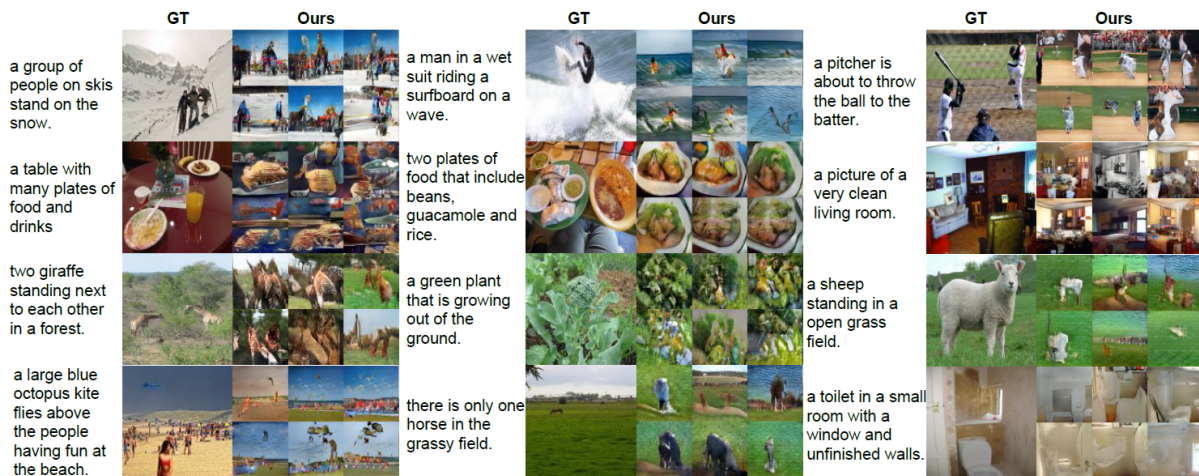


Figure 1.5: Example results of generating synthetic images based on text input done by Reed et al. [4].

A **GAN** is a deep learning network composed of two separate networks, a generative model that produces the image from the data distribution, and an adversarial model that is trained to determine the probability of whether an image is generated or is from the training dataset [22]. **GANs** solve the problem of the inefficiency of using per-pixel loss, relying on an adversarial neural network to develop feature detection, providing more efficient loss values. As the discriminator is trained to better differentiate between *real* and *fake* images, the discriminator's loss is used to update the gradients of the generator to produce more realistic output images. Both models are linked so that progress in one model assists training in the other until the discriminator can no longer accurately discriminate between generated and non-generated images. A traditional **GAN** uses the training data to learn the data distribution, thereafter able to use random noise input to generate a random but realistic image that falls within the feature space of the training data. The general architecture of a **GAN** network can be found in Figure 1.6.

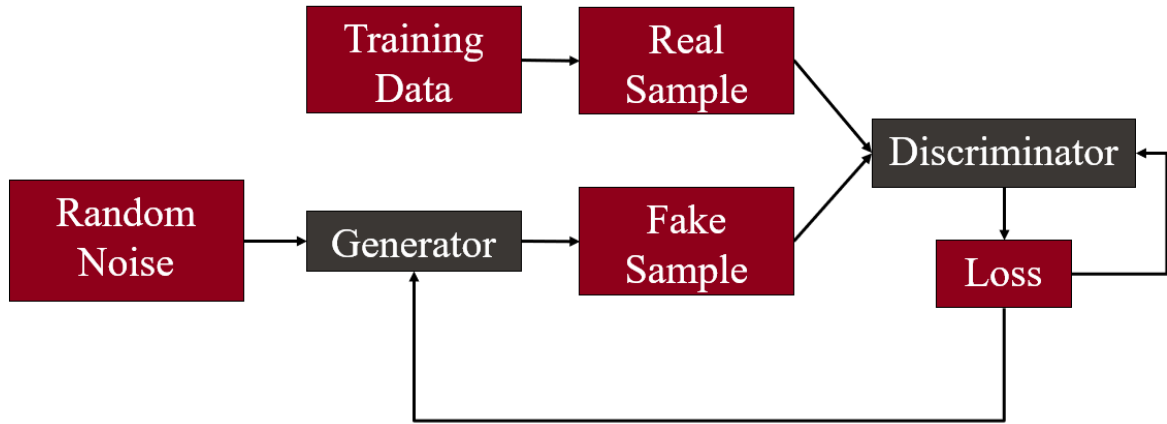


Figure 1.6: Architecture of a typical **GAN** network

CGAN is an extension of **GAN** that uses additional input into the generator and discriminator [23]. In this way, the network is conditioned to produce the desired output given the additional input information rather than through a random noise input. Although there are a few variations to this method, *pix2pix* developed by Isola et al., applies **CGANs** towards image-to-image translation to produce remarkable results, where an input image is used to generate the desired output image [24]. The discriminator in this case must determine if the generated output or the ground truth output is the correct image from the training data.

The success of novel applications using **GANs** has also begun to disseminate into

medicine. Researchers have successfully used GANs to extend scarce chest X-ray datasets by generating random images of common cases as well as the more sought after limited quantity cases [25]. GANs have also been used to solve mature problems such as the 3-D reconstruction of CT data using only 2-D images [26]. Even simple yet elusive problems such as reducing noise in medical images have seen advances from GAN-based approaches [27]. These are a few of the examples that showcase the limitless possibilities available through adversarial networks and their derivatives.

1.3 Motivation

As stated previously, the major limitation in existing approaches to synthetic X-rays is their reliance on input data in the form of 3-D models. An approach to generating X-rays without this data would be useful in non-diagnostic visualization problems. We provide three possible applications where this approach could be used.

The first is in Augmented Reality (AR) surgical applications. AR technologies such as the Camera Augmented Mobile C-arm (CAM-C) system, which use depth cameras to produce visualizations as shown in Figure 1.7, could instead convert video or point cloud data from cameras or sensors into X-rays to create a more intuitive X-ray on X-ray augmentation, similar to live fluoroscopy [5]. Specifically, the CAM-C system uses a recently acquired X-ray which is superimposed on a live point cloud video stream which shows the surgeon’s hands and tools. This is done to allow surgeons to perform tasks using the video feed rather than continuously acquiring additional X-rays. Due to the high noise of point cloud streams, converting the point cloud to X-ray would generate a more natural and intuitive visualization.

A second application can be found in the common problem of 2-D/3-D registration of pre-operative data to intra-operative conditions. By converting intra-operative camera or sensor images into X-rays, it becomes possible to register the image with 3-D data such as a CT image [28]. The registration would provide a relative transformation between the intra-operative pose and the pose in the preoperative data. The successful registration would enable pre-operative data and annotations to align with intra-operative conditions, allowing for crucial planning to carry over. The final application of this approach is for creating flexible simulations for the education of students and the general public. Mobile cameras and other low-cost sensors could be used to create applications that use synthetic X-rays to create interactive and educational visualizations.

In this thesis, we propose a method of generating synthetic X-rays using CGANs to



Figure 1.7: Top-left shows a CAM-C system using a video camera, which produces RGB images that can be used for an AR overlay as shown on the bottom. The top-right shows a CAM-C system with depth cameras. Synthetic X-rays can be generated from RGB images taken from video or depth sensors, and from 2-D/3-D point clouds taken from depth sensors [5].

perform an image translation between RGB images and point clouds to X-rays. We use the hand as initial anatomy to test the first application of CGANs for generating X-rays. We use established methods to build custom software that generates a synthetic dataset used for training. This synthetic dataset is used to train various GAN architectures. We compare the different network architectures and their results when dealing with our specific domain, to establish a baseline for further research. Our goal is to progress synthetic medical image research in the hopes that novel applications that incorporate synthetic images can be developed, ultimately benefiting clinical tasks through reduced radiation and improved medical outcomes.

1.4 Objectives

This thesis developed and evaluated a novel approach to generating synthetic X-rays from point cloud images, RGB images, and RGB images with 3-D point cloud data. The objectives were:

1. Develop a pipeline to generate synthetic X-ray datasets for X-ray image translation.
2. Implement state of the art GANs to translate between 2-D images and X-ray images.
3. Develop a novel architecture that generates more precise synthetic X-ray images by incorporating poses.
4. Examine the effect of introducing 3-D point cloud data to original and novel architectures.

1.5 Technical Contributions

The major technical contribution of this thesis is the development of a novel architecture extended from *pix2pix*. Our architecture is named *pix2xray*, and incorporates the hand pose to increase the quality of generated X-ray images. Our architecture generates more realistic and clear X-rays, and is produces less errors in bone structure.

1.6 Thesis Steps

This thesis accomplished four key steps to establish a baseline for generating synthetic X-ray images from atypical inputs. These steps are:

- Development of a custom dataset generator application capable of generating X-ray images, [RGB](#) images, pose images, and point cloud images in a flexible and fast manner.
- Implementation of general-purpose [CGAN](#) architecture to evaluate point cloud image to X-ray image translation, and the effect of point cloud density on the outcome.
- Development of novel architecture *pix2xray* extended from *pix2pix*, and evaluation of a novel approach compared to general-purpose approaches for [RGB](#) image to X-ray image translation.
- Implementation of *PointNet* to extend *pix2pix* and *pix2xray* and evaluate integrating 3-D point cloud data.

1.7 Thesis Outline

This thesis is laid out into five chapters. In [chapter 2](#), state-of-the-art synthetic X-ray approaches are reviewed, including [Digitally Reconstructed Radiograph \(DRR\)s](#), X-ray data augmentation, text to X-ray synthesis, and X-ray image translation.

In [chapter 3](#) the methodology is presented, beginning with the development of a synthetic data generator, then experiments of point cloud images to X-ray translation, then the development of our novel architecture and experiments on [RGB](#) image to X-ray translation, and finally implementing and experimenting with 3-D point cloud data for X-ray generation.

In [chapter 4](#), the results and discussion of each set of experiments are laid out, beginning with point cloud images, then moving to [RGB](#) images, and ending with 3-D point cloud data.

In [chapter 5](#) the summary of this thesis is presented along with the conclusions and contributions, and avenues for future work are outlined.

Chapter 2

State of the Art

To our knowledge, no work has been done that directly solves the problem of generating synthetic medical X-rays from atypical inputs such as depth and RGB images. There have been significant works over the past few years in regards to generating random synthetic images using GANs primarily for improving limited datasets. The goal of these works has been in achieving higher accuracy in classification, segmentation, and other classical medical image problems. Other works in the area of synthetic medical images also exist that do not use GANs. In this chapter, we examine these state-of-the-art works and overview works that closely aligned with our goals.

2.1 Digitally Reconstructed Radiography

One of the most popular and effective approaches to synthetic X-rays is to generate a DRR from a CT scan [29]. In this method, a CT scan is used as input to generate an X-ray slice by ray-casting through the voxels of the 3-D CT image, estimating the intensity of the final X-ray image through the accumulated hounsfield values of the voxels the ray passes through [30]. There are two benefits to using DRR: the use of patient CT data provides diagnostically accurate images, and a DRR is capable of capturing all soft tissue complexity. DRRs can be effectively used for 2-D/3-D registration problems, where the synthesized X-ray radiographs can be registered to intra-operative radiographs, providing the relative transformation between the modalities [28].

The current state of the art in DRR technology improves upon one of the limitations of the original method; generating a DRR is computationally expensive. Due to DRR using ray-casting, it is possible to overcome this limitation with GPU technology [31]. This

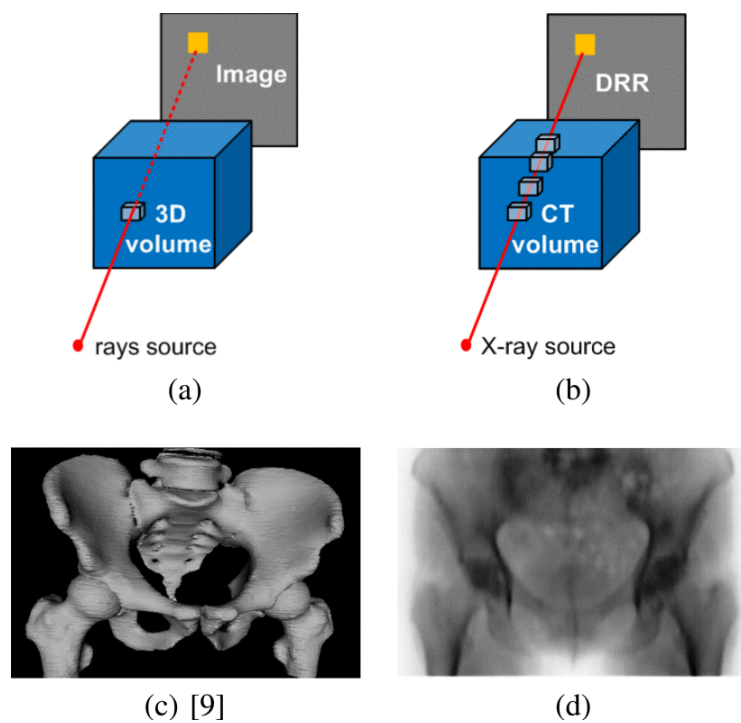


Figure 2.1: GPU-based DRR method created by Dorgham et al., showing a) typical ray-casting method b) DRR raycasting method c) render of the model used and d) sample DRR generated using their algorithm [6].

approach leverages parallel computing and z-buffers in the GPU to reduce computational time. Another recent approach moves away from traditional ray-casting, instead applying light field rendering and computing attenuation fields as a preprocessing step, thus reducing the time to generate a DRR at the expense of memory [32].

The major limitation of using DRR is the requirement that a CT scan is available. As stated previously, X-ray radiation exposure is a serious concern and should be limited and avoided when possible [12]. As a CT scan requires large doses of X-ray radiation, the use of DRR poses a health and safety risk.

2.2 Data Augmentation

Tangent to our research is another area being heavily explored for generating synthetic medical images; the need to improve machine learning accuracy which is inhibited by lack of data. Many medical diseases are rare, and accessible data regarding these diseases is scarce [33]. There is an abundance of healthy patients, but the lack of diseased labels in datasets creates a large class imbalance, which ultimately causes over-fitting or underperformance in trained networks. This reduces the accuracy of networks trained to classify diseases, segment images, and other critical tasks, and due to the need for near perfection in life-saving tasks, machine learning approaches are often dismissed entirely. GANs provide an avenue for solving this limited data problem. Given enough initial data, GANs have proven to be able to synthetically generate new and unique data within that domain [34]. As shown by Frid-Adar et al., liver lesion classification accuracy increased significantly when applying data augmentation through GANs. The same concept was applied to mammography, using the Progressive GAN architecture seen in the previous chapter to create high-resolution mammogram images [7].

There are two major limitations to using GANs to augment existing datasets. The first is that GANs cannot generate data outside the feature distribution it is given. The synthetic image must fall within the domain it is trained on, and will additionally propagate any similarity it sees within the images, even if this is undesirable. The second limitation is that the current state of the art methodology does not capture any patient-specific information. The approaches shown above could not be applied to any task applied to a specific patient.

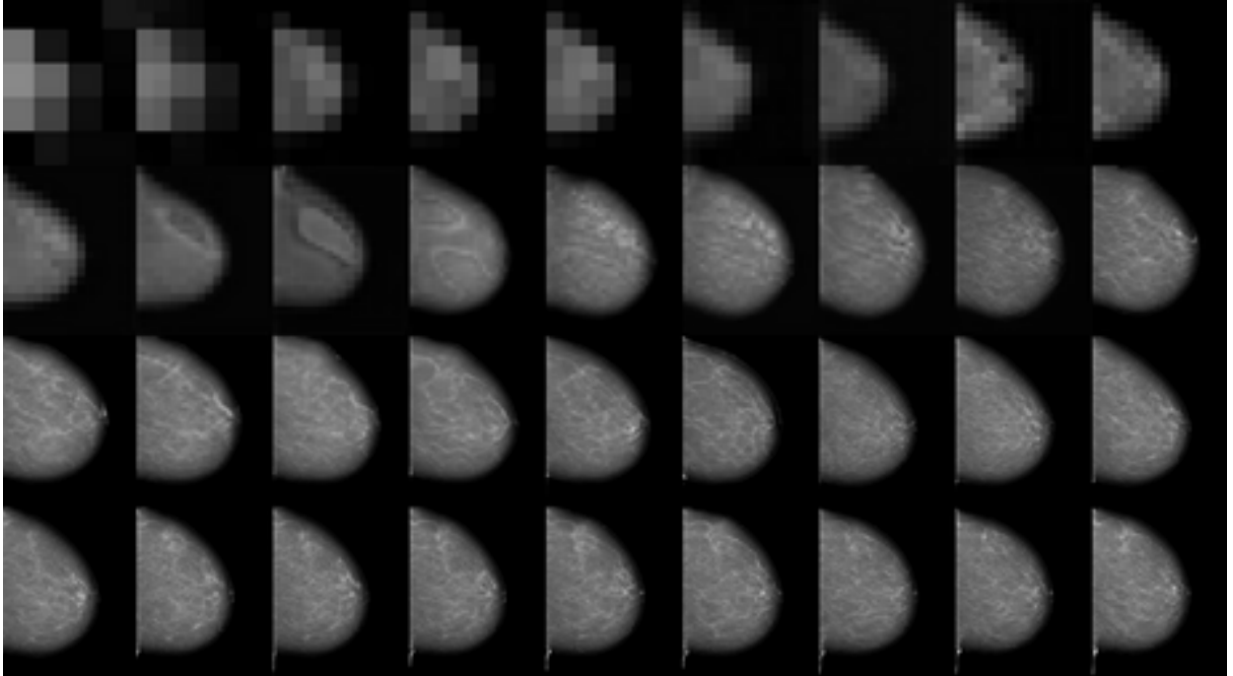


Figure 2.2: Results of applying Progressive GAN towards mammography images done by Korkinof et al., images of higher resolution produced as training progresses [7].

2.3 X-ray Synthesis From Text

Over the past few years, significant research has been published on generating images from text using GANs [4][35]. As a type of atypical input, text provides a significant challenge. A representation must be learned of the variable text input, and the resulting image must be visually realistic and accurate. This area of research has also moved towards medical imaging, with a recent paper by Yang et al., examining X-ray images generated from text taken from radiology reports [8]. This work closely resembles our efforts in generating X-ray images from atypical inputs but instead focuses on text as an input.

They compare their work to multiple other network prominent GAN architectures. *StackGAN* generates high-resolution images from text by implementing multiple generators, which generates an image by incrementally increasing the resolution [36]. *AttnGAN* enhances sub-regions of the image by using specific words from the original text rather than encoding the whole sentence [37]. Lastly, they compare to one of the simpler original implementations by Reed et al. [4]. As seen in Figure 2.3, these networks are trained to generate X-ray images from text descriptions and are compared to *XRayGAN*. The results clearly show the superiority of the approach taken by Yang et al., with care taken on generating frontal and lateral image pairs that are consistent.

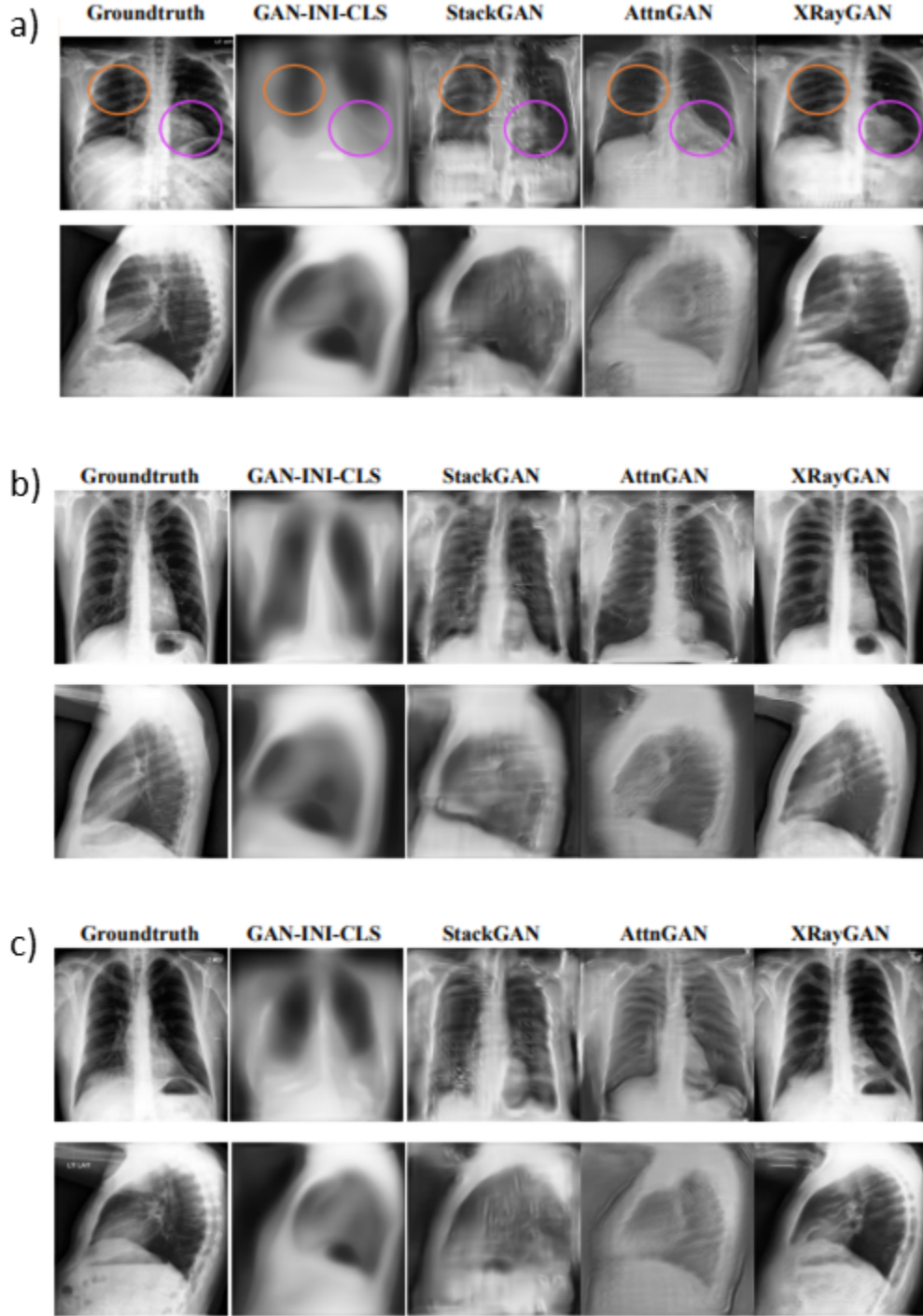


Figure 2.3: Text to X-ray image synthesis conducted by Yang et al., examining popular high-resolution GAN architectures and their novel *XRayGAN* architecture, where X-ray images are generated using text descriptions [8]. Excerpts from the text input for the shown output samples include: a) increased opacity in upper right lobe (orange) and left mid-lung (purple) b) normal lungs with slight degeneration in thoracic spine c) Opacity artifact over right apex and mediastinum and coarse interstitial markings.

The limitations of the *XRayGAN* approach is the restriction to only generate the frontal and lateral view. Although this is effective for their application of generating images for medical students, their architecture focuses heavily on generating these two views and ensuring they are consistent with the same patient. This means that other applications that require flexibility in the direction of the generated image cannot use their approach. Additionally, although their approach might capture patient-specific information, they cannot accurately adjust to small variations in size and shape.

2.4 Medical Image Translation

Research conducted by Teixeira et al. attempts to generate Synthetic X-rays from surface geometry and therefore aligns with the goals of our work [38]. Their approach uses the well established GAN architecture proposed by Isola et al. [24], alongside Wasserstein loss [39], to generate an X-ray image from a surface geometry image. Additionally, they employ another neural network with a *U-Net* architecture to localize landmarks in the X-ray, and a third network is employed to generate an X-ray image from the landmarks. These last two networks are trained in tandem so that the iterative loop of training causes convergence in both networks. The goal is two-fold: (1) they facilitate the generation of synthetic X-rays from surface geometry and (2) using the landmarks, additional random X-rays can be generated by randomly adjusting the landmark locations, thus making this work applicable to quickly creating large datasets through data augmentation.

The results shown by Teixeira et al. in Figure 2.4 prove that generating synthetic X-rays using GANs is possible. Their use of landmarks is a creative solution for augmenting data and provides benefits towards applications such as device positioning. The downside to this approach is that surface geometry is not always readily available. To utilize this approach, a depth camera would need to be used, and some form of image to surface geometry technique would need to be applied. Therefore our research is a critical next step, paving the way for other inputs so that the application of this work does not become limited.

Another related work is the research from Tmenova et al., where image translation between synthetic X-rays and real X-rays is showcased [40]. In this work, *CycleGAN* architecture is applied to convert synthetic X-rays generated using simulation software to look like X-rays taken intra-operatively [41]. Although this work does not directly relate to our efforts in generating synthetic X-rays from atypical inputs as our work uses synthetic X-rays to create a dataset, incorporating methods such as this could bridge the gap between

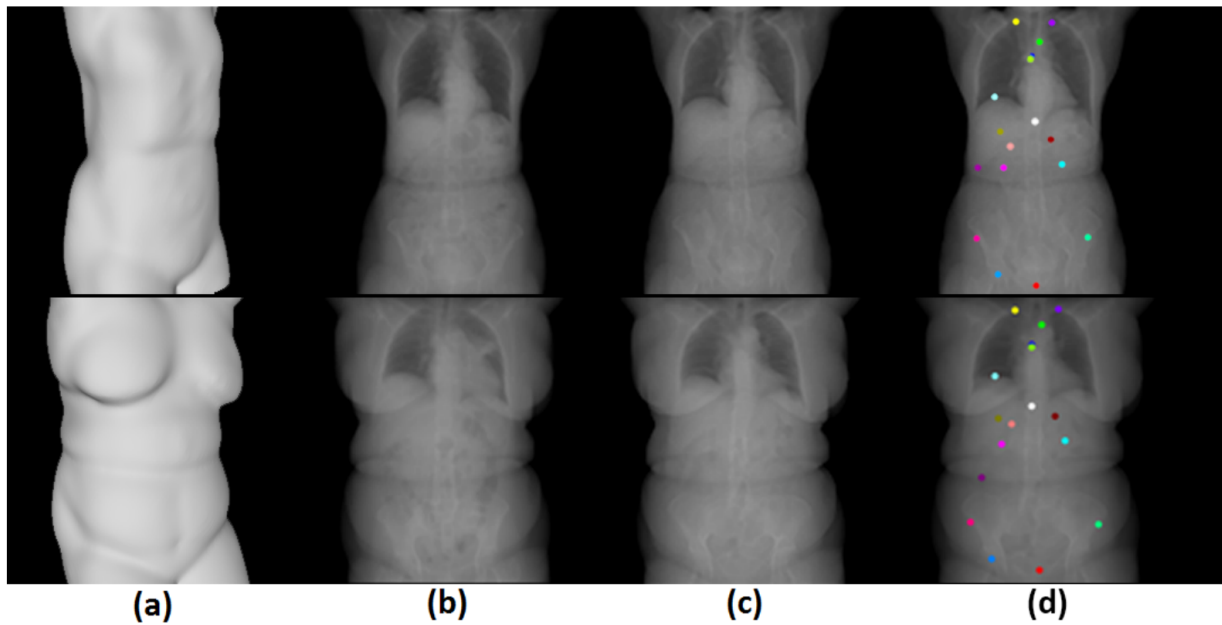


Figure 2.4: Generating synthetic images from surface geometry accomplished by Teixeira et al., a) surface geometry b) ground truth c) generated image and d) generated body markers to allow for data augmentation

synthetic outputs and realistic X-rays.

Chapter 3

Methodology

3.1 Synthetic Data Generator

The first step is to acquire a large, flexible, and relevant dataset. To our knowledge, no public datasets exists for the purpose of X-ray image translation. Although general X-ray data can be found publicly, our project requires that the X-ray images be paired with input images of an anatomy, point clouds, and even poses. No dataset exists that meet this criteria and no application exists that can generate images in this format. Due to this restriction, as well as difficulty in creating a dataset with real X-rays, we chose to create a simulation environment and produce our own synthetic dataset. Our test application will focus on human hand anatomy.

Our generator application uses meshes to generate synthetic X-rays as well as other paired image types based on the need of the user. The meshes used in our environment were created using Blender, a 3-D modeling software [42]. We created a hand model in Blender that is adapted from a publicly available accurate human hand model. The original model as seen in Blender is shown in Figure 3.1, with the outer skin mesh, inner bone mesh, and the pose armature visible. To conform the hand to a natural hand pose, we chose to base the hand pose from the open-source [New York University \(NYU\)](#) hand pose dataset [43]. This dataset of hand poses contains 80,000 natural hand poses taken using multiple Kinect sensors. We sample a few of these hand poses, taking poses of different variety, and apply them to our hand mesh through the pose armature. This allows us to create a small dataset of hand meshes in various natural poses.

The hand models are imported inside our custom generator application. The generator application is a simulation environment built in OpenGL using C++. The software uses

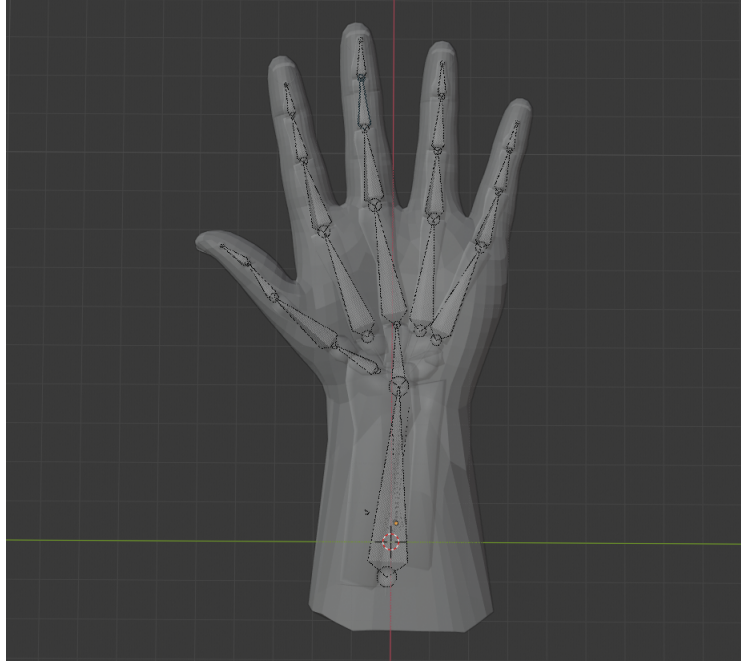


Figure 3.1: Hand model as seen in Blender, with the inner and outer meshes visible as well as the armature which is used to make poses.

other open-source libraries such as GLFW, ASSIMP, and DearImGui, as well as other smaller libraries to create a flexible application for simulating X-rays and generating image data. A screen-shot of our generator application is showcased in Figure 3.2. The X-ray simulation itself is accomplished through the open-source library *gVirtualXray* created by Vidal et al., and provides GPU-based real-time X-rays generated through ray-tracing [2]. We chose this software as the X-ray images generated using *gVirtualXray* are within a one percent margin of error for X-ray estimation when compared to real ground truths [18]. Furthermore, the GPU-based approach performs 60 times faster than the state of the art CPU-based libraries. These two factors allow us to create a simulation environment with a live X-ray simulation for rapid image creation, and the ability to generate synthetic X-rays whose authenticity has been extensively validated. *gVirtualXray* is limited to an object with two meshes, an inner and outer mesh, but is capable of manipulating the X-ray factors and material properties. The output size, resolution, image quality, and beam parameters can all be manually adjusted for specific applications.

The generator produces data through an iterative randomization process. At every iteration, the hand model is translated and rotated within a certain bound. After images and X-rays for one hand pose are saved as a Bitmap, the next hand pose is imported and the process continues, cycling through each of the natural hand poses. Translation

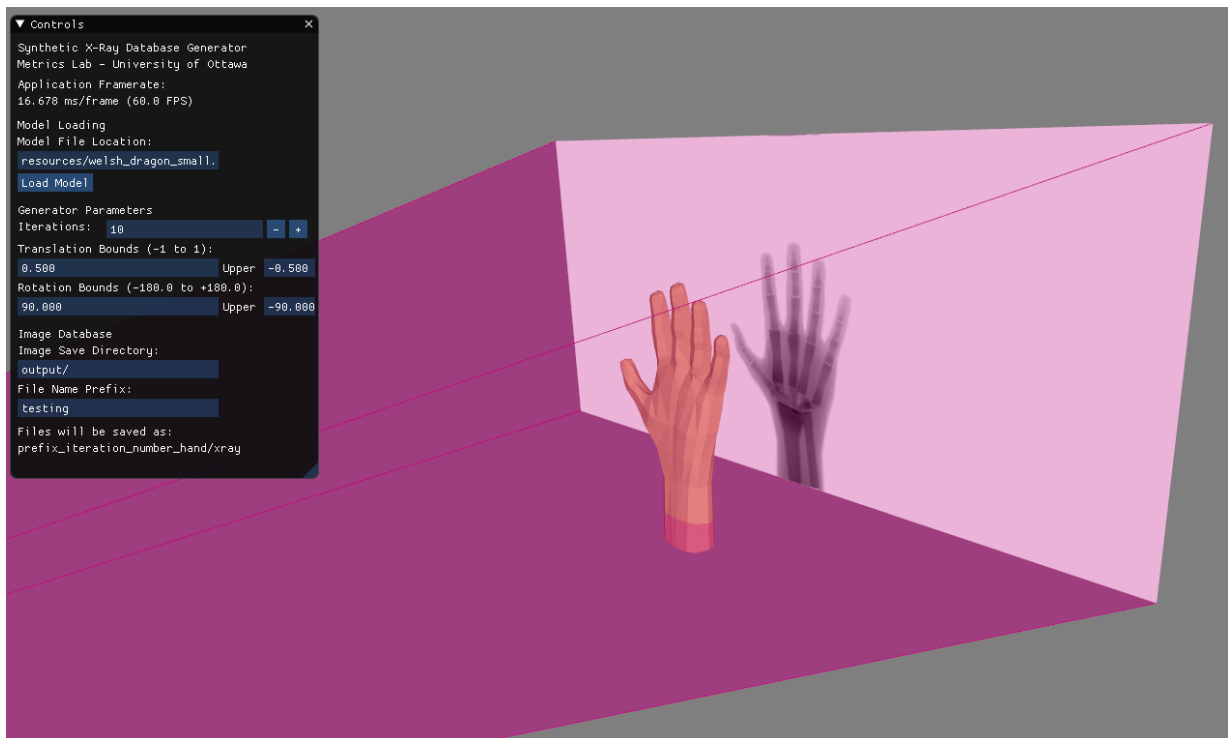


Figure 3.2: Screen-shot of our synthetic data generator application, which hosts a real-time X-ray simulation and the ability to randomly create large datasets.

is randomized in the plane of the detector and is bound within the screen space so that the hand is never partially occluded by the edge of the detector, this ensures the entire hand is always visible on screen. Rotation is bound within 30 degrees in all planes so that the hand remains palm-side toward the user. This is done to limit the scope of this project to focus on one side of the hand, reducing the complexity of dealing with both sides of the hand. We choose the hand, rather than another anatomy, due to the original motivation of translating the surgeon’s hands found in the surgical [AR CAM-C](#) system. This synthetic data generator approach allows us to create a very large and varied dataset of 1000 images within minutes through a safe and accurate manner for use in our non-diagnostic visualization project.

3.2 Point Cloud Images

To meet Objective 2 of the thesis, point clouds are generated using surface sampling through the Monte Carlo sampling algorithm found in MeshLab [\[44\]](#), and different densities of point clouds are created to compare the effect of the number of points to the resulting X-ray image quality. To avoid sampling inner bone structure, Ambient Occlusion is used to filter out the occluded faces of the inner mesh. The point cloud images are created by rendering the point cloud as vertices in OpenGL inside the generator and saving the pixels of the final output. The point cloud is superimposed with the mesh that is generating the X-ray so that both the point cloud and the X-ray maintain the same transformation.

Six point cloud densities examined by varying the number of points sampled throughout the hand mesh. The six densities are 50K, 40K, 30K, 20K, 10K, and 5K. These six densities provide a variety between high and low densities to allow for analysis of density towards generating X-rays. Densities lower than this would be too sparse to generate an X-ray, and higher densities are not practical as depth cameras do not have the technology to provide extremely high-density point clouds. From these six densities, 1000 images are generated per density by taking each hand pose and randomly transforming to generate 100 images before moving to the next pose. Additionally, each dataset includes a separate test dataset of 200 image pairs generated separately, through randomization no image is the same between the training and test datasets. The overall work-flow employed for generating and training with point cloud images is highlighted in [Figure 3.3](#).

For training, we implement *pix2pix* by Isola et al., a general-purpose [CGAN](#) for image translation problems that has proven to be effective in multiple domains [\[24\]](#). *pix2pix* utilizes a *U-Net* architecture, following the typical encoder-decoder network architecture

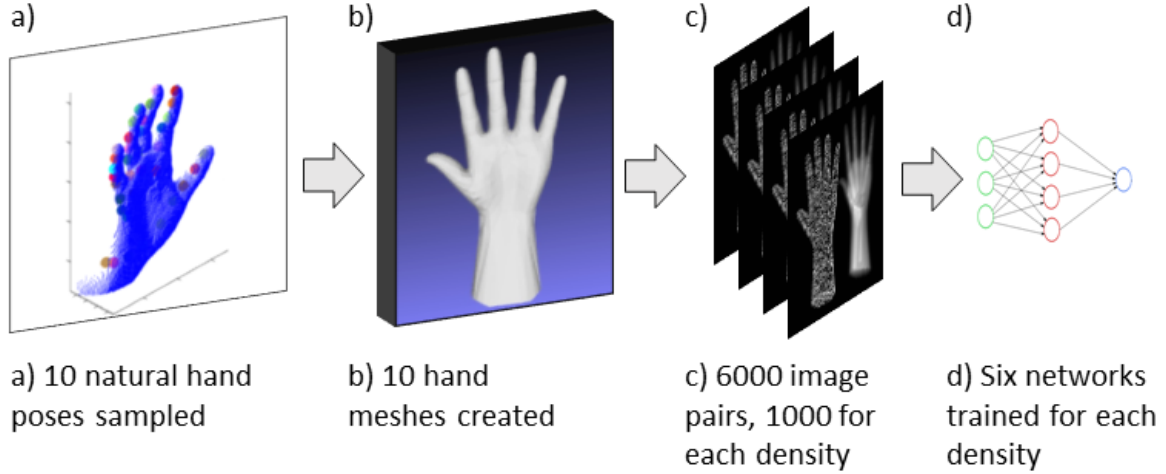


Figure 3.3: Overall work-flow for training the point cloud to X-ray translation networks starting with a) sampling 10 natural hand poses from the [NYU](#) hand pose dataset b) creating 10 meshes with the generated poses c) generating the six synthetic datasets of 1000 image pairs through randomization of each pose 100 times in the generator and d) training the six networks.

of multiple down-sampling layers followed by an equal number of up-sampling layers to re-create the image. Each of the eight down-sampling blocks is composed of a convolution, a batch normalization, and a leaky ReLu activation. Each of the eight up-sampling blocks is composed of a deconvolution, a batch normalization, and a ReLu activation. The first three up-sampling blocks also include a drop-out layer with a frequency rate of 0.5.

Additionally, skip layers are added between mirrored layers of the encoder and decoder to maintain sharpness in images, this is done by concatenating between the mirrored outputs of the down-sampling and up-sampling blocks. A 70x70 *PatchGAN* is used as the discriminator to penalize incorrect images based on patches of the full image. 70x70 is the most optimal size as it produces sharper results and better discrimination scores [24]. The discriminator uses the same layer structure of convolution, batch normalization, and leaky ReLu activation, but structured to produce the patch-wise discriminator output values. Figure 3.4 shows a high-level diagram of the *pix2pix* network.

Training of the Generator G and the Discriminator D to learn the mapping between input x and output y is done through the equation:

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y} [\log D(x, y)] + \mathbb{E}_{x,z} [\log (1 - D(x, G(x)))] \quad (3.1)$$

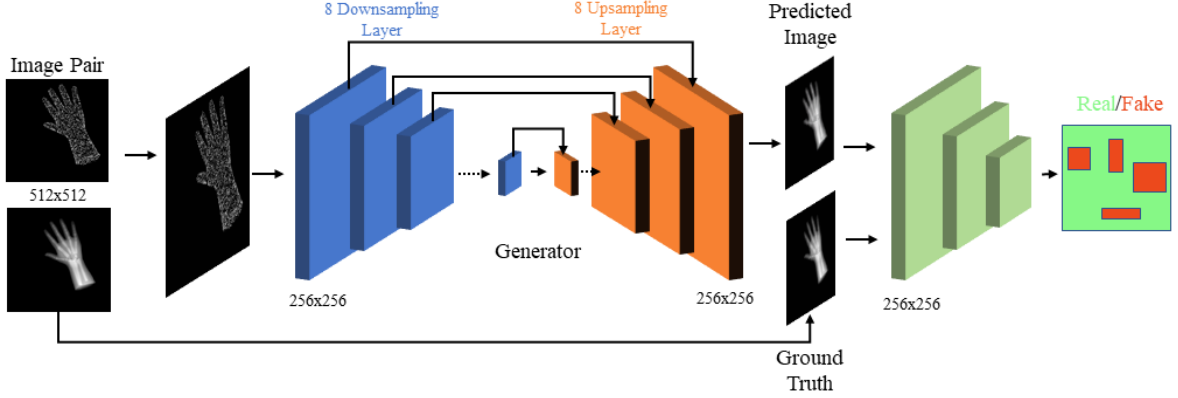


Figure 3.4: Overview of the pix2pix GAN architecture.

The generator tries to minimize this loss, whereas the discriminator tries to maximize this loss, and thus competition is born between them. Additionally, Isola et al., have proven that incorporating L_1 loss towards training the generator avoids blurry results [24].

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y} [\|y - G(x)\|_1] \quad (3.2)$$

Thus, the combination of these two functions provides the function for training the generator:

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \quad (3.3)$$

A lambda value of 100 is used as the established value in literature for best results [24]. Random initializers are used to sample from a normal distribution of mean 0 with a standard deviation of 0.02, and TensorFlow’s Adam optimizer is implemented with a learning rate of 0.0001 and a β_1 value of 0.5. Each image is originally 512x512 in resolution and is scaled down to 256x256 to match the network input. Each point cloud image set is trained separately, for a total of 140 epochs.

3.3 RGB Images and Pose to Resolve Occlusion

After the successful results of our point cloud to X-ray experiments using a general-purpose architecture, we move forward to attain Objective 3 of the thesis by utilizing a new type of input, RGB images. In this set experiments, we change input modalities, but also

try to solve the major limitation from the point cloud experiments; occluded hand poses cause inconsistencies in the output image. To solve this, we propose a modified network architecture that incorporates the pose to overcome errors caused in occluded scenarios. To accomplish this, we first added the functionality in our generator to produce image triplets instead of pairs, producing an image of the hand pose at every iteration.

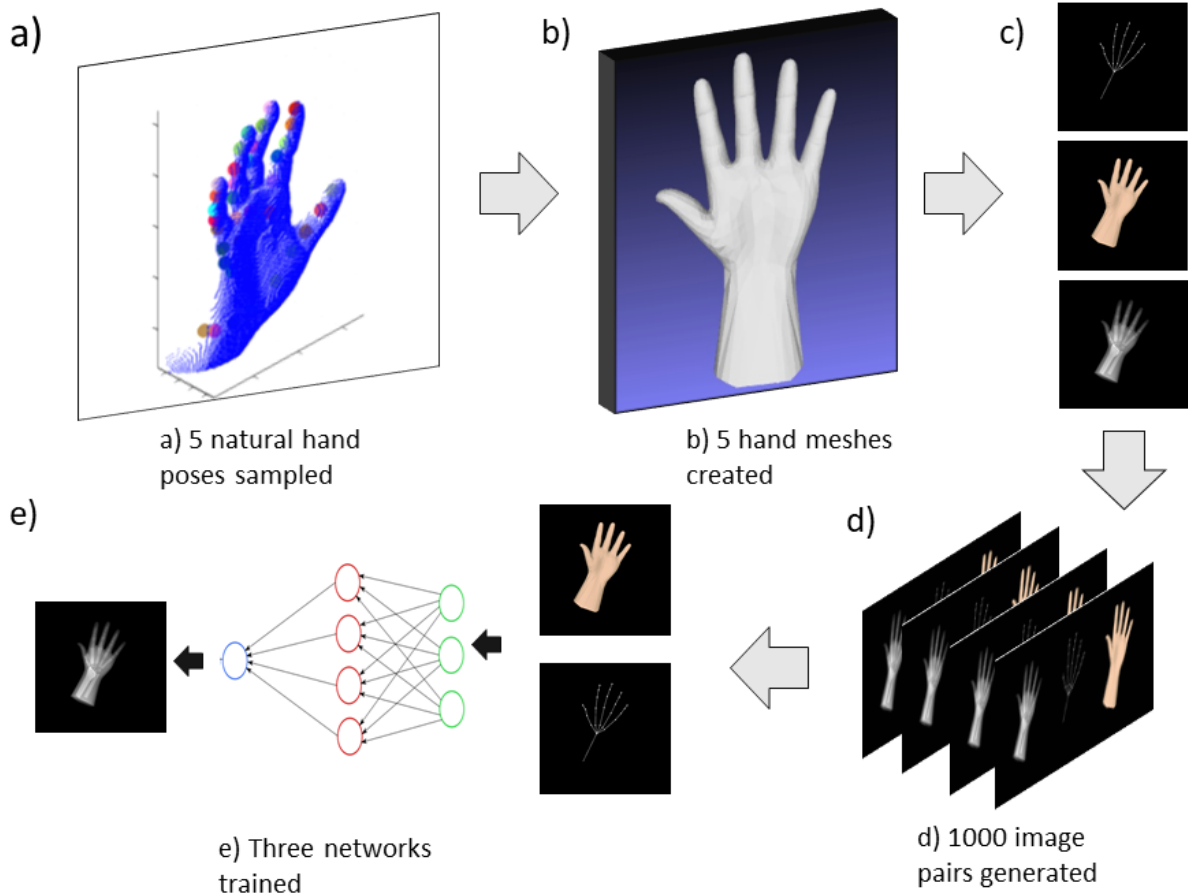


Figure 3.5: Overall work-flow for training the RGB image to X-ray networks starting with a) sampling 5 natural hand poses from the NYU dataset b) generating 5 hand meshes c) importing into generator to create image triplets d) generating a dataset 1000 image triplets through randomization of each pose 200 times and e) training three networks: *pix2pix*, *CycleGAN*, and *pix2xray*.

The hand pose is generated using a creative solution. In Blender, thin cylinders are added to the hand mesh at the centreline of each finger and through the palm. The cylinders are joined by small spheres at joint locations. The cylinders and joints - which we refer to as a pose mesh - when viewed independently, visually look almost identical to a hand pose skeleton. This pose mesh is tied to the pose armature, and when each

natural hand pose is applied to the model, the respective pose mesh is also simultaneously created. This pose mesh is imported into the generator alongside the hand mesh, and at every iteration, the same transformations are applied to the pose mesh, and an image is created of the pose by hiding the other mesh and saving a Bitmap. This method ensures that the poses we generate are accurate and are not created through external sources, such as hand pose estimation neural networks, which would introduce a possible source of error. With the various hand poses and random transformations, we create a dataset of 1,000 image triplets by randomly transforming a single hand pose 200 times and capturing the data before moving to the next pose. 500 triplets are additionally generated for validation, 100 of each hand pose. From the test set, we also isolate 200 images of only occluded cases to be used as a separate test for analyzing occluded cases. The overall workflow for generating the image triplets can be seen in Figure 3.5 and a sample of image triplets can be seen in Figure 3.6.

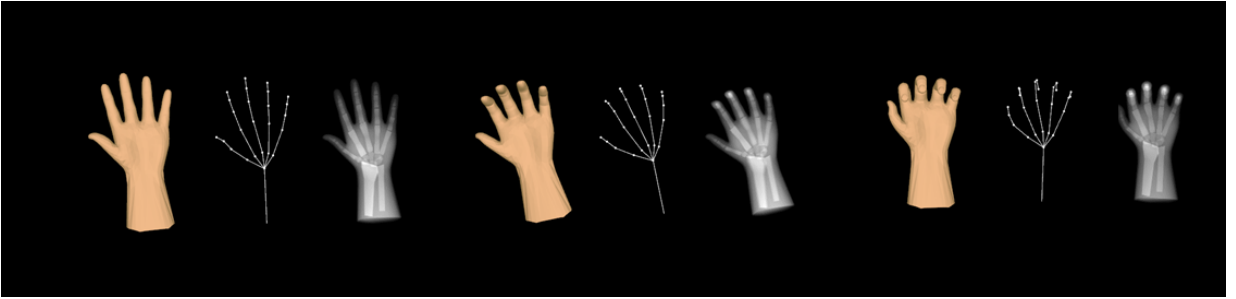


Figure 3.6: Sample of image triplets created through the generator, each triplet consisting of a RGB image, pose image, and X-ray image.

pix2xray is our modified architecture that expands on the *pix2pix* architecture to generate clearer results, especially in occluded cases. Occluded cases are a problem as the overlapping of hand features corresponds to less visual information for the network to work with, thus leading to poorer results. We define occluded cases as those in which portions of the hand and fingers are hidden due to the bending of the fingers. To solve the occlusion problem, we incorporate the pose of the hand itself into the network to assist the network in creating a more accurate representation. To accomplish this we create a stacked network composed of two individual networks. The stacking of encoder-decoder networks has been previously successfully applied in literature for tasks such as human pose estimation [45]. We apply the same theory to our problem, using the first network to create an intermediate 2-D image of the hand pose, which is then used as input alongside the original image to the second network.

The first network is built in the typical encoder-decoder style, with eight blocks of down-sampling and eight blocks of up-sampling. The blocks use layer structures similar to the *pix2pix* architecture discussed previously. The output of this network is then stacked with the original RGB image to create a $256 \times 256 \times 6$ image, which is used as input for the second network which is identical to the *pix2pix* implementation discussed previously. The networks are trained independently, the joint estimator is trained first, then the X-ray generator is trained using the trained joint estimator to create the pose image. Figure 3.7 shows the architecture of our modified network compared to the original *pix2pix*.

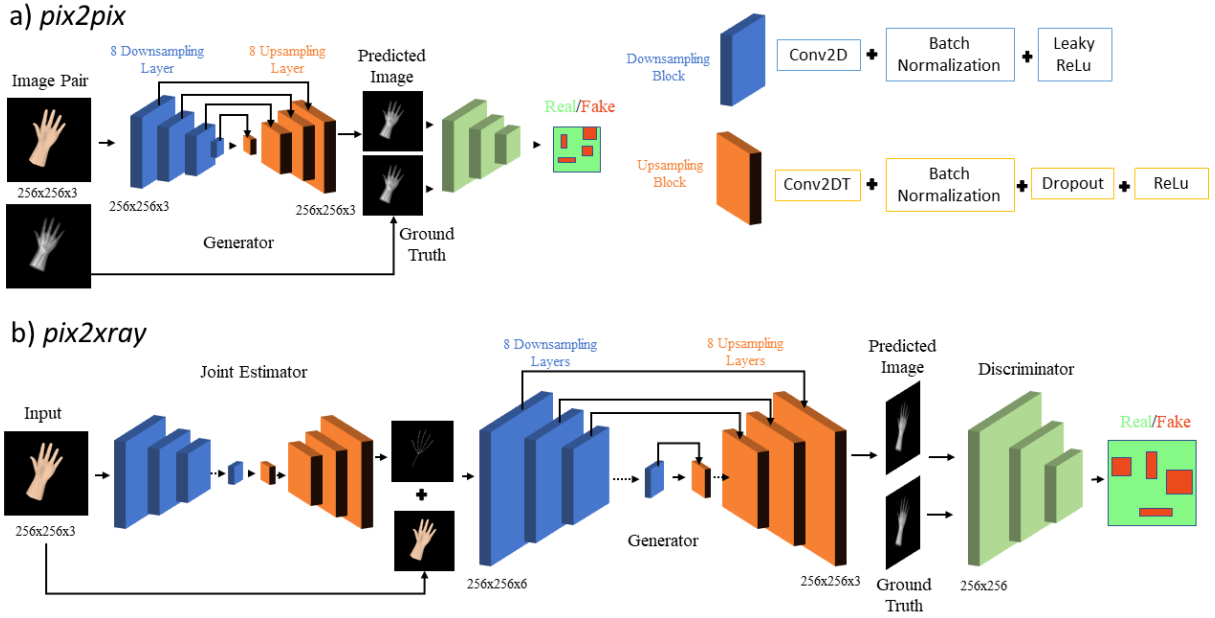


Figure 3.7: Overview of our modified architecture *pix2xray* in comparison with the original *pix2pix* architecture.

When training our new architecture, we compare our results with a network based on *pix2pix* as seen in the previous section, as well as a third network based on the *CycleGAN* architecture [41]. *CycleGAN* is a general-purpose approach that provides unsupervised unpaired image-to-image translation, in contrast to *pix2pix* which is a supervised paired image approach. *CycleGAN* uses two generators and a discriminator for each generator. This approach does not require paired images and uses the concept of cycle consistency to ensure that one generator can map between domain X to domain Y, and a second generator can accomplish the inverse mapping. To accomplish this, the networks are motivated to ensure that for any input x , if it passes through the first generator G which maps between

$X \rightarrow Y$ and then passes through the second generator F which maps the inverse $Y \rightarrow X$, then the result should be x . This is called forward cycle consistency, and the inverse starting with input y would be backward cycle consistency. The incentive to ensure cycle consistency is through the loss function:

$$\mathcal{L}_{cyc}(G, F) = \mathbb{E}_x [\|F(G(x)) - x\|_1] + \mathbb{E}_y [\|G(F(y)) - y\|_1] \quad (3.4)$$

With this addition, the adversarial loss and cycle loss can be added to create the final objective:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN}(G, D_Y, X, Y) + \mathcal{L}_{GAN}(F, D_X, Y, X) + \mathcal{L}(G, F) \quad (3.5)$$

Both generator networks must work to minimize this loss, while the discriminators work to maximize, and so the competition is created in while training. The general architecture of *CycleGAN* utilizes two convolutional layers, nine residual blocks, and two deconvolutional layers. These residual blocks are composed of two convolutional layers, with a connection between the input and output to allow for residual information to be passed on, which has been proven effective in generating results [46]. Additionally, the network uses instance normalization as opposed to batch normalization, and the discriminator is a 70x70 *PatchGAN* which penalizes images at the level of individual patches as opposed to per-pixel or per-image basis. The architecture overview of *CycleGAN* is displayed in Figure 3.8

All networks are trained from scratch using our data, and do not use any pre-trained weights. Between the networks, we maintain similarity in training so that the comparison between qualitative and quantitative results can be drawn. For each network, we use the same dataset of 1000 images, with 200 images used for validation. The original image sizes of 512x512 are scaled down to 256x256 for each network. We run each network for 140 epochs, which is a sufficient number for the convergence of the model as shown in other image translation experiments [47]. Each epoch consists of a batch size of the entire dataset. Before training steps, data is randomly cropped and mirrored to ensure the same image is never used twice. We stay true to the original implementations of the papers to establish a solid baseline for this first approach to image translation. A lambda value of 100 is used for *pix2pix* and *pix2xray*, and a value of 10 for *CycleGAN*. Random initializers are used to sample from a normal distribution of mean 0 with a standard deviation of 0.02, and TensorFlow’s Adam optimizer is implemented with a learning rate of 0.0001 and a β_1

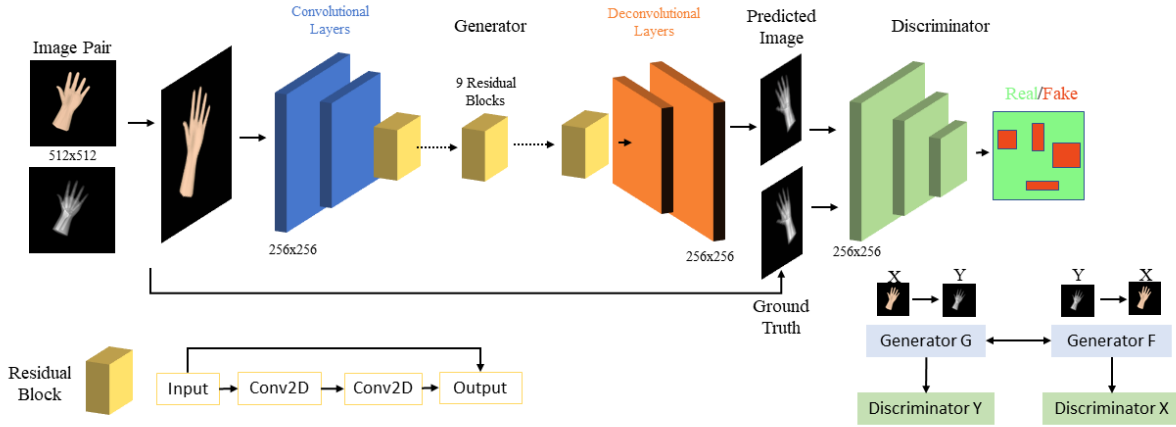


Figure 3.8: Overview of CycleGAN architecture.

value of 0.5.

3.4 3-D Point Cloud Data

We move beyond 2-D inputs and examine the effect of 3-D data on generating synthetic images. To meet Objective 4 we specifically examine 3-D point clouds, defined as a set of point coordinates representing the surface of an object. Point clouds are unordered by nature, the opposite of ordered data such as an image. This also means that rotation and translational variation between point clouds is irrelevant, as the relationship between points is the only meaningful information. Although certain tasks such as object recognition can use raw point clouds as input [48], the use of raw point clouds to generate images is not possible due to the inability to capture the contextual information of the entire point cloud. The most powerful method to incorporate point clouds in generating images is to simultaneously use the point cloud with a paired image. In this way, the input image provides contextual information, such as the location of the object in the image, color, and background, and the point cloud input provides structural information about the object to better generate an accurate shape [49]. We employ the same method, a branched network that accepts two inputs, an input image, and an input point cloud, and outputs an X-ray image.

The point clouds are generated using the same method as seen in the previous point cloud experiments. The points are sampled from the surface faces of the mesh, ignoring bone structures. A Monte Carlo sampling method is employed, with 40 thousand samples

taken. We use 40 thousand points as the previous experiments showed strong results for this point density. Separate point clouds are created for each pose, and when training and testing the point cloud for that pose is used as input.

To implement the branched network, the *pix2pix* architecture and our own *pix2xray* architecture are adapted to accept and process the point cloud input. We ignore *CycleGAN* in this set of experiments as results made it clear that *CycleGAN* cannot produce high quality results. To accomplish the point cloud information processing, we implement the state of the art architecture introduced by Qi et al., in their *PointNet* model [50]. This adaptation is necessary, as the typical *U-net* architectures have no way of handling point cloud data adequately. Their work introduces a layer architecture that processes unordered point cloud input into a global feature vector of size 1024x1. This global feature vector can then be used in further layers to perform classification and segmentation. We employ this global feature vector to concatenate with the innermost layer of the *U-Net* architecture. Thus both the point cloud and image input information are passed into the up-sampling portions of the *U-Net* architecture to generate the image. To align with the *U-net* architecture, we modify the *PointNet* architecture’s final layers to ensure an output global feature vector of 512x1, so that it aligns perfectly with the already existing architecture used in previous experiments. The architecture of the branched network can be seen in Figure 3.9. The *PointNet* branch uses a transform block that creates a matrix that is multiplied with the prior input. Multilayer perceptron blocks are introduced between the transform blocks for further convolutional processing, and the final output is the global feature vector formed after max pooling.

We train three networks with point cloud inputs. The first two networks use the branched version of the *pix2pix* architecture, examining the effect of 3-D input to both RGB images and point cloud images. The third network is the branched version of our own proposed architecture *pix2xray*. We compare these three networks to their original counterparts to assess the impact of including 3-D point cloud data, as well as the difference between the type of paired input image used. We run each network for 140 epochs. Each epoch consists of a batch size of the entire dataset. Prior to training steps, data is randomly cropped and mirrored to ensure the same image is never used twice. A training set of 1000 images and a validation set of 500 images is used as showcased in the previous section. A separate validation of 200 images that only include occluded images is also used for validation for further analysis.

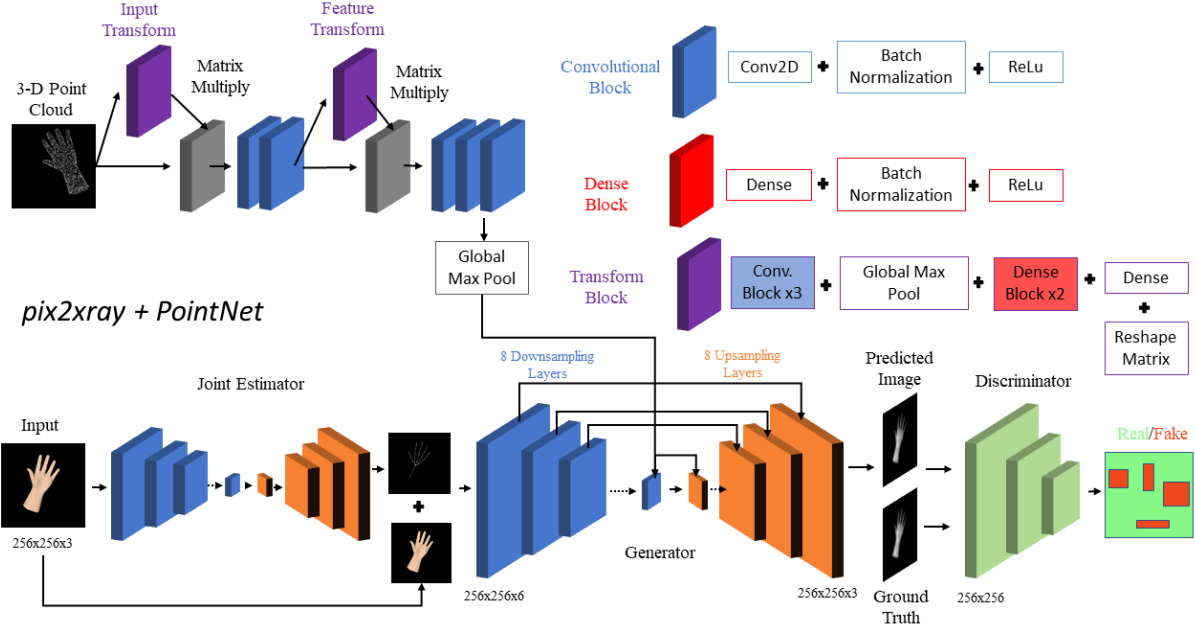


Figure 3.9: Architecture overview of our modified *pix2xray* including the *PointNet* branch for point cloud processing.

3.5 Validation

To validate our results we compare the predicted images in a test dataset to the ground truth X-ray image generated from the [GPU](#) ray-tracing library. We use the classic pixel accuracy metric which tells us the mean absolute difference between all pixels, as shown through the equation:

$$M = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y} \right| \quad (3.6)$$

Where y_i represents a pixel from the ground truth and $\hat{y}_i$ represents a pixel from the predicted image. Although this metric is beneficial as a rough estimate of similarity, noise, blur, and artifacts can heavily impact this evaluation. To provide an additional metric that can overcome this limitation, we use the [Structural Similarity Index Metric \(SSIM\)](#) [51]:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (3.7)$$

This metric examines the luminance, contrast, and structural properties of two images x and y , and has been shown to better assess the similarity between images over the traditional methods of pixel accuracy and mean error. Each component is independent of the other, and therefore variations in one component such as luminance will not impact structure. A value of 1 indicates identical images, whereas 0 indicates no similarity. We use a sliding filter size of 11x11, and K1 and K2 values of 0.01 and 0.03 respectively as suggested by the original paper. [Root Mean Squared Error \(RMSE\)](#) and [Peak Signal to Noise Ratio \(PSNR\)](#) are also measured to provide additional metrics, as these metrics are the most common measurements for image similarity as shown in previous studies [40]. Lower [RMSE](#) values and higher [PSNR](#) values indicate better image similarity. The implemented equation for [RMSE](#) is:

$$RMSE = \sqrt{\left(\frac{1}{n}\right) \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (3.8)$$

The equations used for [PSNR](#) are:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2 \quad (3.9)$$

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \quad (3.10)$$

Chapter 4

Results and Discussion

This chapter includes results that were published in a conference and journal publication, with a third publication pending.

- Results from Section 4.1 were presented at the International Conferences of the IEEE Engineering in Medicine and Biology Society in 2020.

M. Haiderbhai, S. Ledesma, N. Navab, and P. Fallavollita, “Generating X-ray Images from Point Clouds Using Conditional Generative Adversarial Networks,” presented at the *International Conferences of the IEEE Engineering in Medicine and Biology Society*, Montréal, Jul. 2020.

- Results from Section 4.2 were published in the International Journal of Computer Aided Radiology and Surgery in 2020.

M. Haiderbhai et al., “pix2xray: converting RGB images into X-rays using generative adversarial networks,” *Int J CARS*, Apr. 2020, doi: 10.1007/s11548-020-02159-2.

- Results from Section 4.3 are currently under submission.

4.1 Point Cloud Images

This section examines the results of working with point cloud images of different densities to generate X-ray images. Table 4.1 compares the [RMSE](#), [PSNR](#), and [SSIM](#) values of the networks trained on each separate point cloud density dataset. The comparison between the difference in point cloud densities and the ability to generate realistic X-rays can be

drawn from the results. Low **RMSE** values, alongside **SSIM** values closer to one and high **PSNR** values indicate greater image similarity.

Figure 4.1 highlights multiple sample inputs from the 50K point cloud density dataset, and their resulting prediction compared to the ground truth. Figure 4.2 highlights the difference between the predictions made by the six different networks. Figure 4.3 shows failure cases that qualitatively show a significant lack of similarity due to increased noise and lack of correctly defined bone structure. A summary of the qualitative observations can be found in Table 4.2.

Table 4.1: Evaluation of image similarity between the predicted images of different point cloud density inputs with their respective networks and the ground truth.

Point Cloud Density	RMSE		SSIM		PSNR	
	Mean	SD	Mean	SD	Mean	SD
50K	0.073	0.01	0.921	0.015	22.85	1.74
40K	0.068	0.01	0.925	0.015	22.53	1.71
30K	0.073	0.01	0.923	0.016	22.94	1.94
20K	0.068	0.01	0.926	0.015	23.54	1.57
10K	0.081	0.01	0.915	0.019	21.98	1.61
5K	0.087	0.02	0.909	0.019	21.41	1.94

Table 4.2: Summary of qualitative differences between point cloud density predicted X-rays.

Point Cloud Density	Summary
50K	Small inconsistencies visible but plausible prediction
40K	Small inconsistencies visible but plausible prediction
30K	Small inconsistencies visible but plausible prediction
20K	Small to medium inconsistencies visible but plausible prediction
10K	Large inconsistencies visible in bone structure and shape
5K	Large inconsistencies visible in bone structure and shape

When comparing the quantitative metrics, higher point cloud densities show better results compared to the lower densities. As seen in Table 4.1, 50K, 40K, 30K, and 20K point densities show similar results, with **RMSE** values below 0.08, **SSIM** above 0.92, and **PSNR** above 22. Compared to these four high point density datasets, the two low density datasets at 10K and 5K show higher **RMSE** at 0.081 and 0.087, lower **SSIM** at 0.915 and 0.909, and lower **PSNR** at 21.98 and 21.41 respectively. These values indicate that the two lower point cloud densities are on average less similar to the ground truths. This result

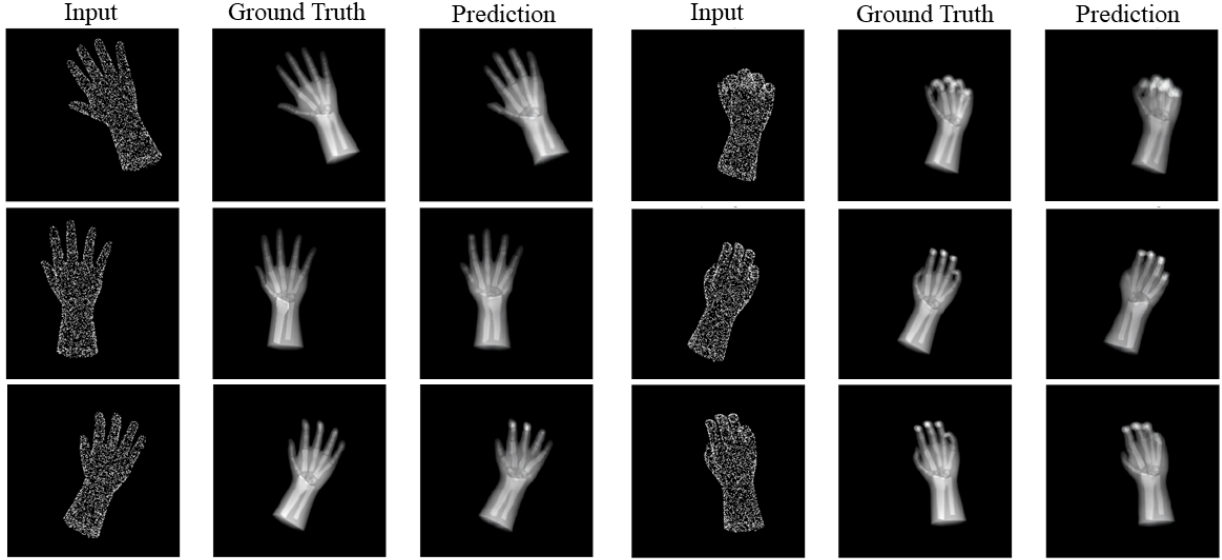


Figure 4.1: Sample of predicted X-rays generated using the 50K point cloud density network and dataset.

is to be expected, as lower densities provide limited information for the network to work with. The lower densities have sparser point clouds, and there is a larger distance between visible points on the image. Therefore, the network must estimate more pixels between points causing more noise and less structural similarity.

The qualitative results showcased in Figure 4.2 highlight how these quantitative values translate to the final image quality. Both 10K and 5K show more distortion in the bone structures compared to the higher density models as seen in Figure 4.4. Although almost all examples show slight imperfections in the predicted X-rays, the lower density predictions have immediately noticeable inconsistencies, thus detracting from the goal that the images should be plausible. The 20K density also has a large bone abnormality in the image as seen in Figure 4.5, but quantitatively it should perform similar to the higher densities. This is an example of how generative networks can be inconsistent when generating images, even when giving similar inputs. This type of GAN error has been documented in the literature and is a key concern when working with GANs [52]. Due to the ambiguous nature of deep learning, it is difficult to identify why a network could produce inconsistent results, but a likely factor could be a lack of variety in the training data, or simply an inability to create a perfect mapping between input and output. Our future efforts in providing additional data when generating images hopes to reduce this type of consistency error. With two different types of input, the generator has more information to better create a mapping

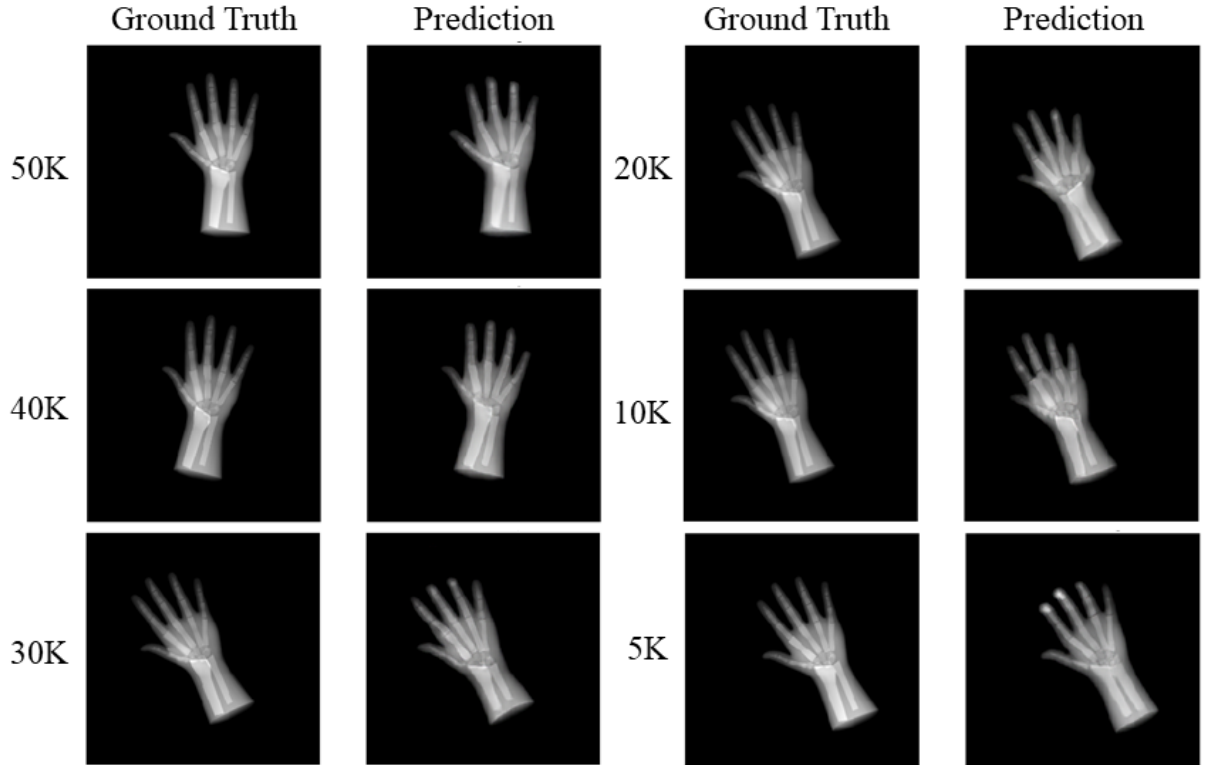


Figure 4.2: Sample of predicted X-rays from each of the six different networks using a similar pose. The first column of each pair shows the ground truth X-ray, the second column shows the predicted X-ray generated from a point cloud image input of the density specified on the left.

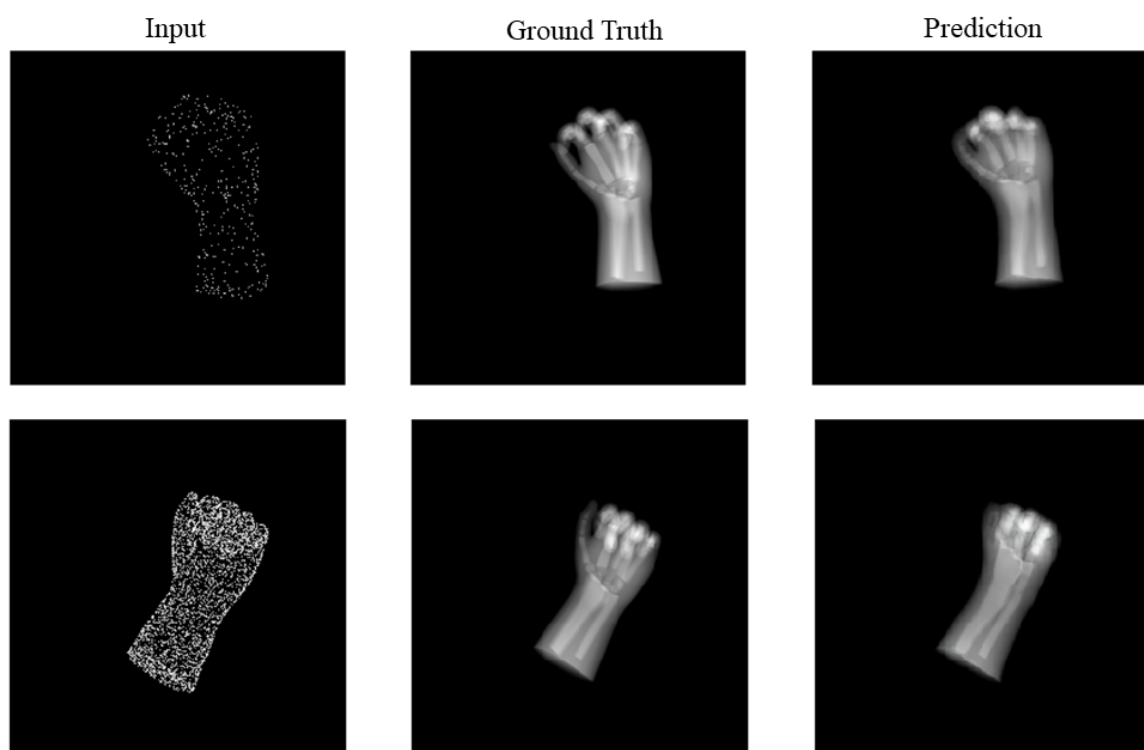


Figure 4.3: Sample of low 5K (top) and high 50K (bottom) point cloud density input that produces a visible failure case.

between the two domains.

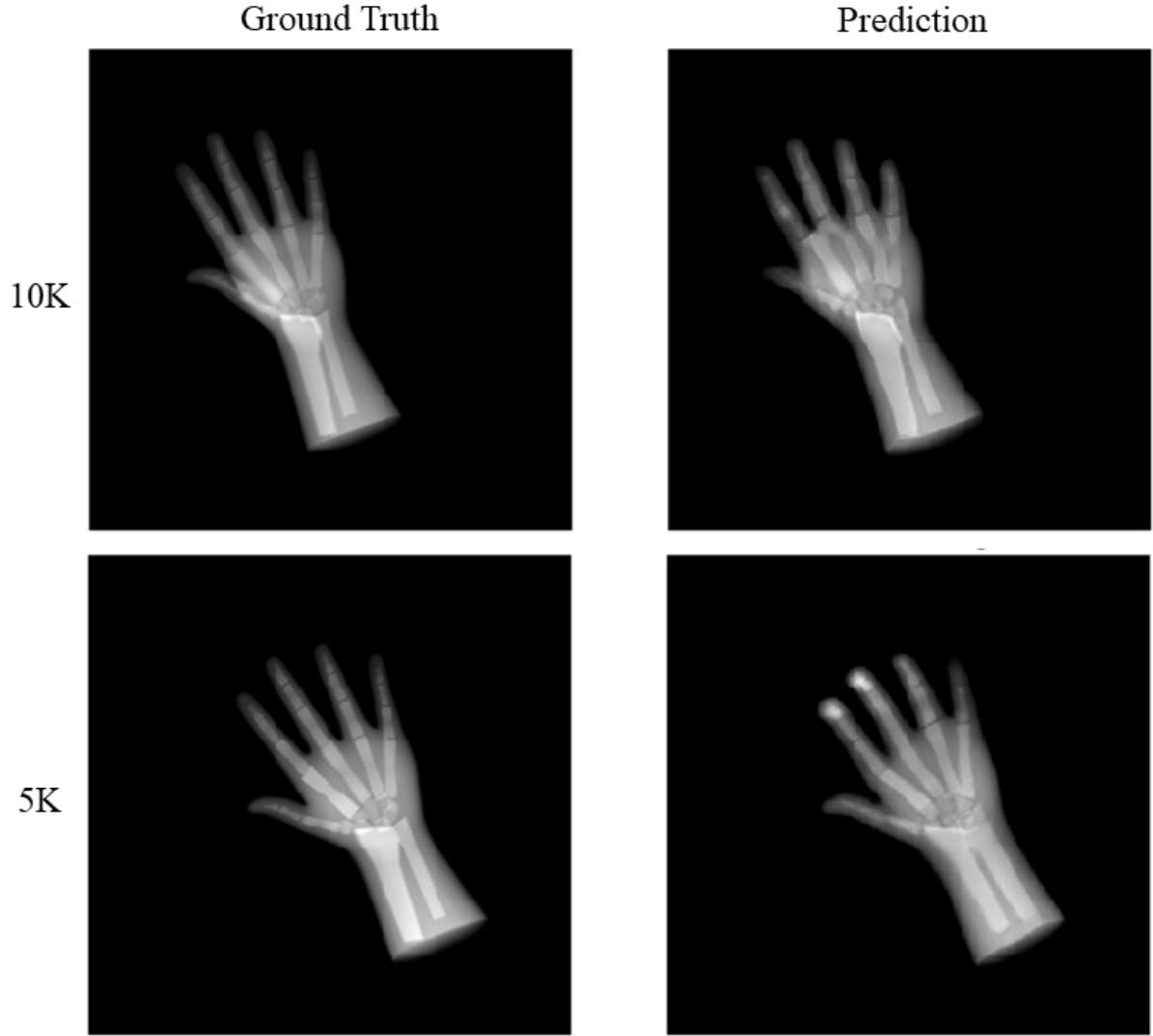


Figure 4.4: Enlargement of 5K and 10K predicted X-rays to visualize the inconsistencies in these low density inputs.

It is also evident that overall, the models fail to accurately translate X-ray images in situations with high occlusion. As seen in Figure 4.1 and further seen in Figure 4.3, when fingers begin to occlude each other and other portions of the hand, the resulting X-ray begins to lose well defined and correctly placed bone structures. The reason for this loss of quality is due to the overlapping points caused by the occlusion scenarios, and the loss of clear information results in the network unable to generate accurate overlapping bone structures. *pix2pix* provides a general-purpose approach to image translation problems and thus can be applied to problems such as ours in medical imaging. Although this architecture

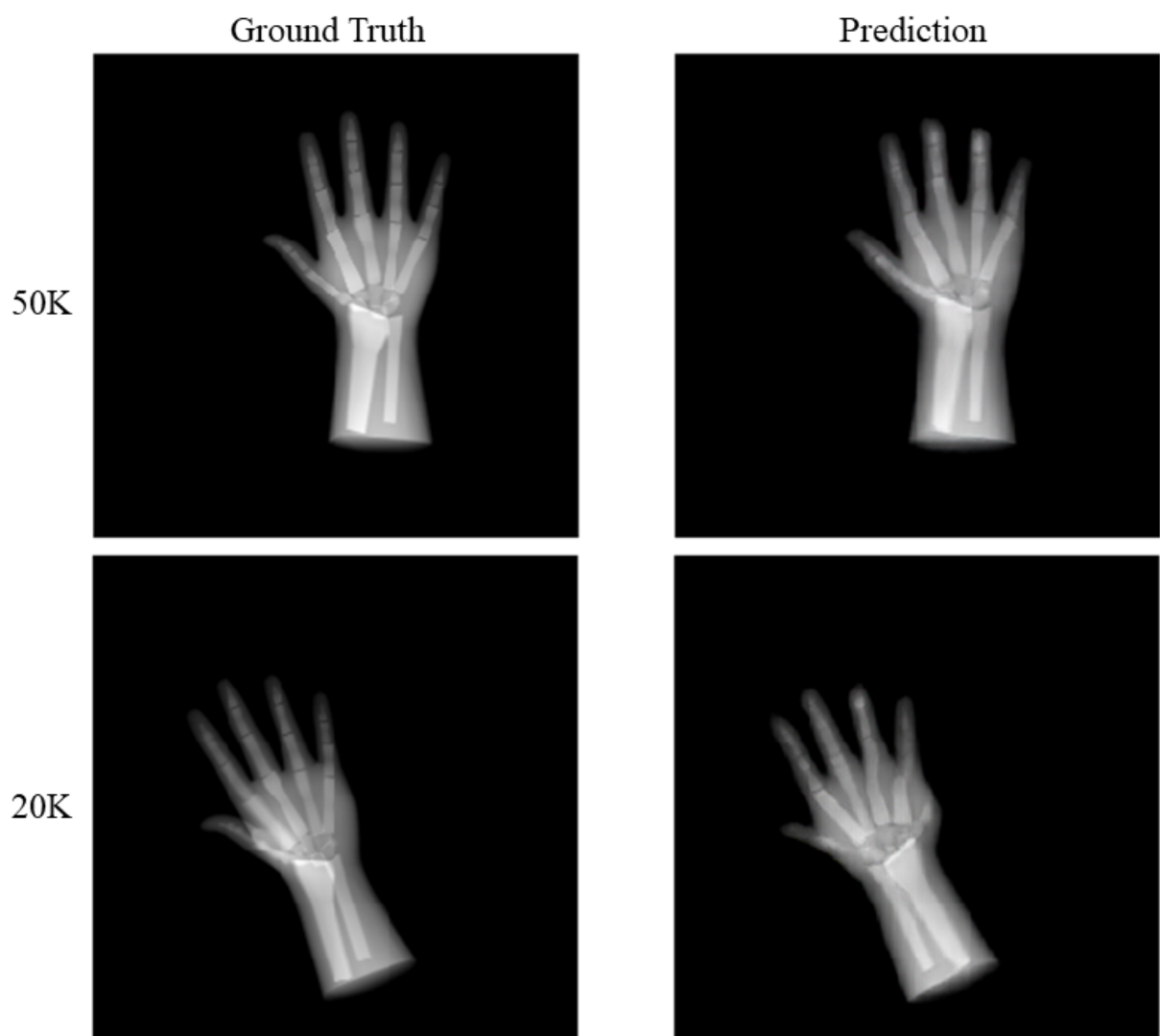


Figure 4.5: Enlargement of 50K and 20K predicted X-rays to visualize the inconsistencies and difference between high density inputs.

provides remarkable results, it shows its weakness in situations with occlusion. As seen in the next set of experiments, an architecture that incorporates two sets of input information provides a better solution to combat occlusion.

Our approach assumes a complete point cloud of the anatomy under study. This is not always the case; in real-life, applications using depth sensors only provide points that are facing the camera. Typical [Red Green Blue and Depth \(RGB-D\)](#) sensors work exactly in this manner, providing an image that only has points corresponding to the surfaces facing it. Additionally, the camera could provide any density of point cloud, there is no guaranteed standard. To solve this concern of partial point clouds as well as variation in point cloud density, it may be necessary to have a two-step process, where point clouds of any variation are first processed using a [CGAN](#) or similar network to produce intermediate data in the form of an image or pose. Then the X-ray is formed using this intermediate data, which would be the same regardless of the type of point cloud used as input. Similar two-step approaches have been used in the past to recover occluded objects in images, where first the occluded shape is recovered, and then the texture and colors are generated [53]. Other limitations of these results include the lack of clinical data used in validation. We hope to introduce validation using real X-rays in the future as we bring these ideas towards clinical applications.

These findings highlight the possibility of generating synthetic X-rays from atypical inputs. The sparse nature of point clouds makes it a difficult input to work with, but these initial results show that X-rays can be generated even in the face of difficult inputs. Additionally, point cloud are not an accessible input format, as they require specialized technology in the form of depth sensors. In the next section we examine the next set of results which deal with [RGB](#) images, an image format that is widely accessible through hand-held camera technology. We also examine our own method to limit errors in occluded images, and examine a third network architecture that uses unsupervised learning rather than the supervised methods seen in this section.

4.2 [RGB](#) Images and Pose

This section analyses the different approaches taken to generate X-ray images from [RGB](#) image inputs. In Table 4.3, we compare the measured similarity metrics of the generated X-ray to the ground truth for the three networks, *CycleGAN*, *pix2pix*, and our own *pix2xray*. The metrics are gathered from a test dataset built separately from the training data, which includes 5 separate poses with 100 randomizations to provide a test dataset of 500 images.

Table 4.4 examines the metrics generated from a separate test dataset that only includes occluded images.

Figure 4.6 visualizes the resulting predicted images for four test cases, and Figure 4.7 visualizes more difficult cases due to large amounts of rotation and occlusion of the hand to see how the networks handle these cases. A summary of the qualitative observations can be found in Table 4.5. We determine these cases to be difficult or failures qualitatively with the help of one end-user to examine the images. We provide clinical data comparison in Figure 4.8 through X-rays taken from a bone phantom using a mobile C-arm device. Equivalent predicted X-rays taken from identical perspectives in our generator are shown to compare our predicted outputs to real X-rays.

Table 4.3: Evaluation metrics of image similarity between the two general-purpose architectures *CycleGAN* and *pix2pix*, and our own approach *pix2xray*.

Network	Pixel Accuracy	SSIM	RMSE	PSNR
CycleGAN	50.28	0.930	0.0948	20.95
pix2pix	78.81	0.955	0.0552	25.38
pix2xray	78.27	0.967	0.0450	27.12

Table 4.4: Evaluation metrics of image similarity in unique test dataset with only occluded cases.

Network	Pixel Accuracy	SSIM	RMSE	PSNR
CycleGAN	47.97	0.934	0.0926	21.12
pix2pix	66.98	0.957	0.0579	24.97
pix2xray	82.24	0.967	0.0460	26.93

Table 4.5: Summary of qualitative differences between outputs of networks using RGB input.

Network	Summary
CycleGAN	Blurry image but large inconsistencies in bone structure
pix2pix	Clear image but large inconsistencies in plausible bone structure and shape
pix2xray	Clear image with clear and plausible bone structure and shape

The results indicate that our *pix2xray* architecture outperforms both *pix2pix* and *CycleGAN* in producing images with higher similarity and clearer results. This superiority is seen both in the quantitative and qualitative results gathered. As seen in Table 4.3

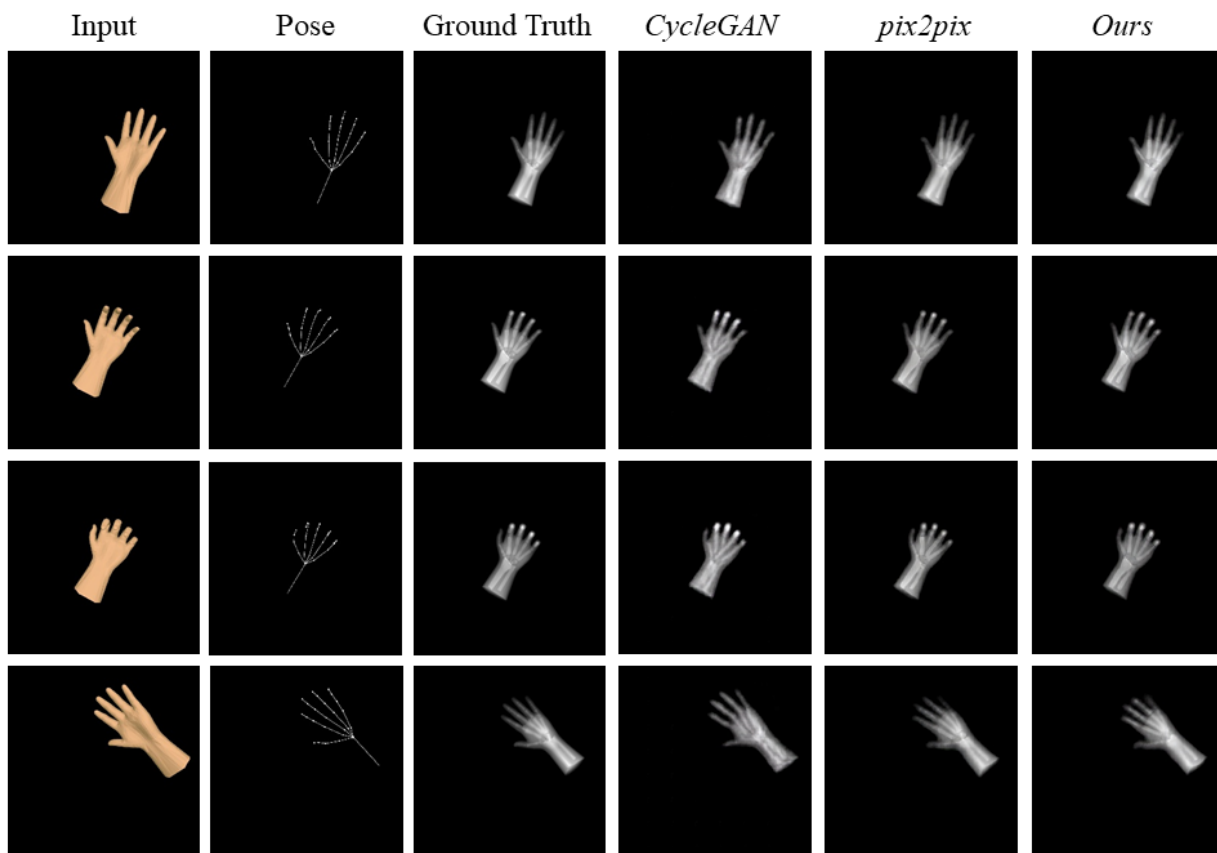


Figure 4.6: Comparison of predicted X-rays between *CycleGAN*, *pix2pix*, and our network *pix2xray* using sample inputs.

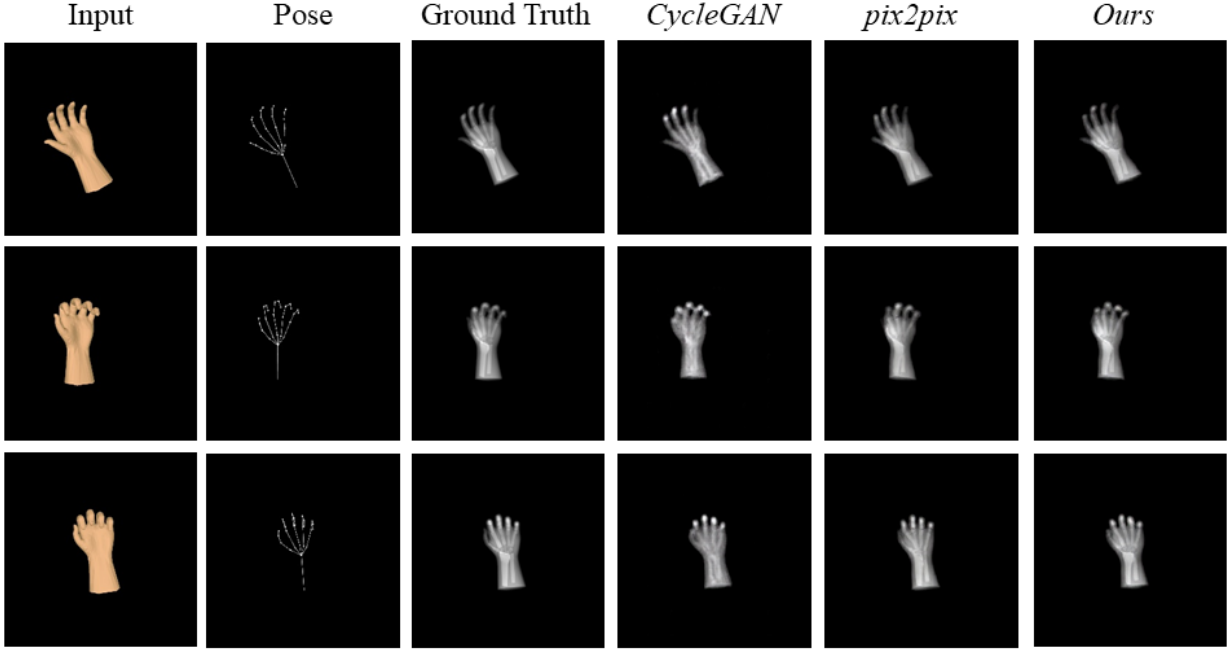


Figure 4.7: Comparison of predicted X-rays when handling occlusion between *CycleGAN*, *pix2pix*, and our network *pix2xray* using occluded sample inputs.

Our approach has the highest [SSIM](#) at 0.967, the highest [PSNR](#) at 27.12, and the lowest [RMSE](#) at 0.450. These findings assert that our approach performs better than *CycleGAN*, and also marginally better than *pix2pix*. The pixel accuracy of our approach is not higher than *pix2pix*, which we attribute to the fact that our images are slightly darker as a whole compared to *pix2pix*, most likely carried over from the second black pose image we use to generate our image. This also reaffirms our use of [SSIM](#) as a superior metric, as [SSIM](#) factors in luminance, contrast, and structure independently, thus avoiding the misleading results that pixel accuracy leads to. Our [SSIM](#) value is higher than both *pix2pix* and *CycleGAN*.

The results can be visualized by examining Figure 4.6, where we see that our approach produces clear bone structure, and correctly defines the bone structure in a plausible manner. *CycleGAN* produces blurry images and bone structures that are extremely incorrect, which explains the low similarity metrics it achieved. Although the results of *pix2pix* are clear, compared to our approach it fails to always place the bone structures correctly. This can be seen clearly in the third row with the fused and half missing metacarpal bones, or in the first row with the bones of the fingers blending in with the nearby soft tissue. Figure 4.9 enlarges the images to clearly see the difference between networks.

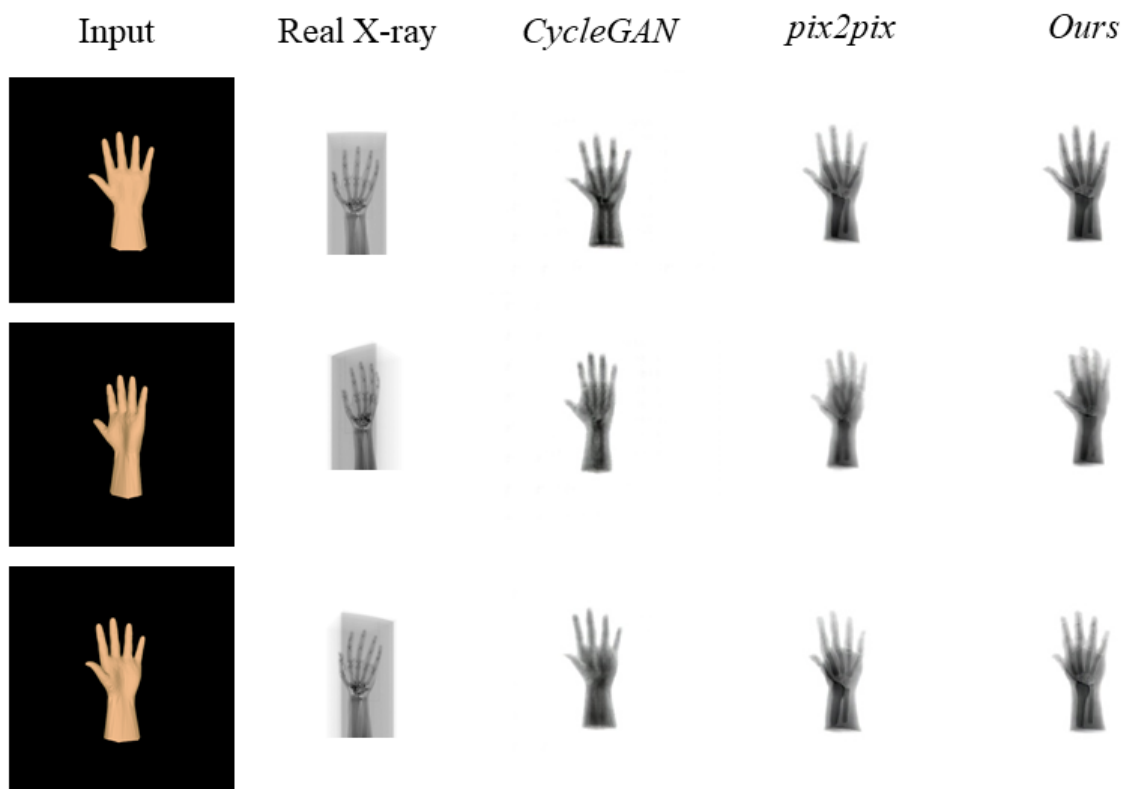


Figure 4.8: Comparison of predicted X-rays to real X-rays taken from a bone phantom, using similar poses generated in our generator.

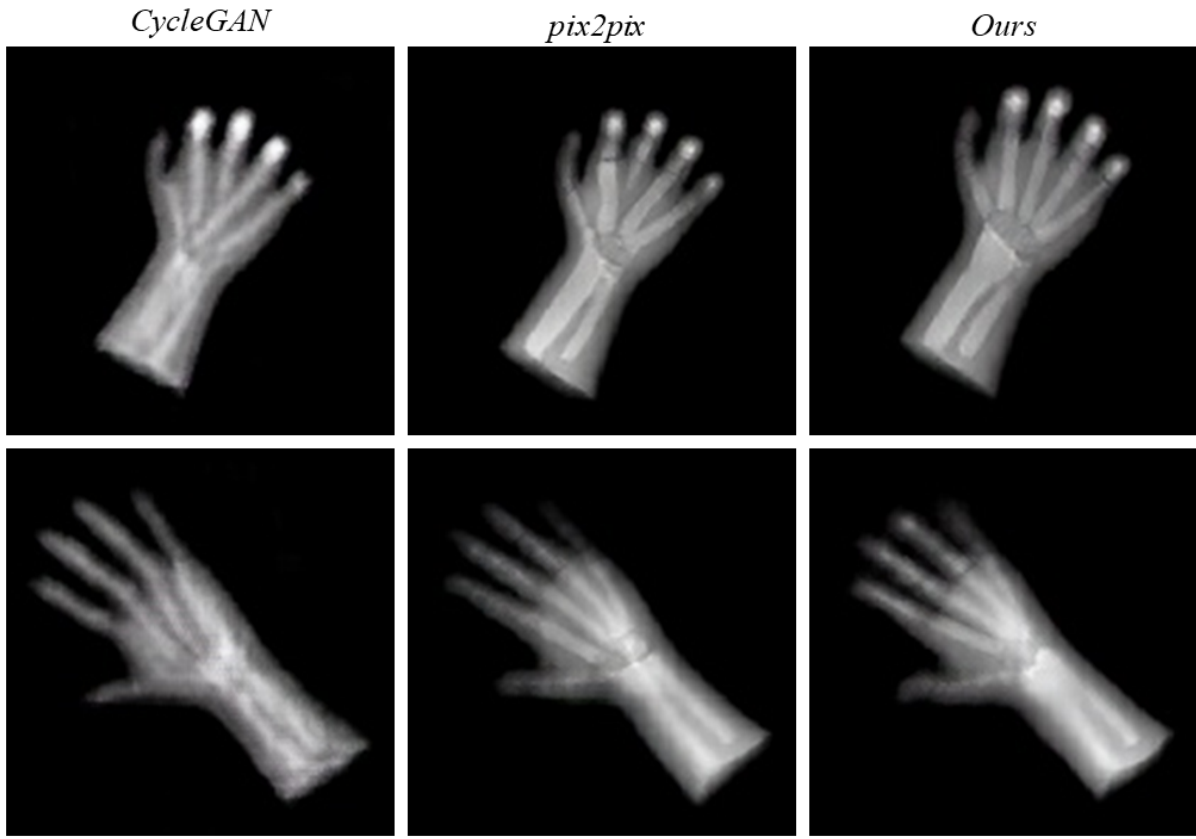


Figure 4.9: Enlargement of predicted X-rays generated from *CycleGAN*, *pix2pix*, and our network *pix2xray*, to visualize the inconsistencies.

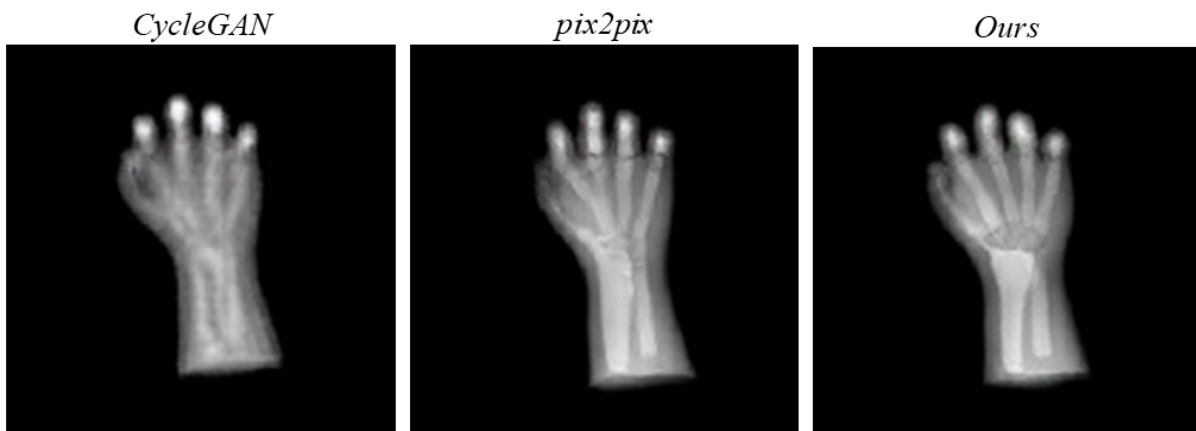


Figure 4.10: Enlargement of predicted X-rays generated from *CycleGAN*, *pix2pix*, and our network *pix2xray*, from the occluded test set to visualize the inconsistencies.

The goal of our approach was to better handle occluded cases, where fingers and bones are overlapping and occluding each other, by incorporating the hand pose as a secondary input produced from the same input image. In the simpler open hand examples, every pixel provides the information for the corresponding X-ray pixel, but in the occluded cases, the same pixel must provide information for multiple bones. To examine our results in isolation, we tested the resulting networks on a dataset of only occluded examples, where the poses had curled fingers. As seen in Table 4.4, introducing this occluded dataset causes both *CycleGAN* and *pix2pix* to drop in similarity as seen in the pixel accuracy. The other metrics seem to get better though, but this is to be expected. In the occluded images, the hand takes up less space and the remaining black background takes up more space, and so the images in one way are indeed more similar due to the similarity drawn from the black portion of the image. In this scenario, pixel accuracy provides the best analysis, as within the remaining pixels of the hand, the difference between pixels is greater, which causes a large decrease in pixel accuracy. Future research will aim to solve this by relying more on user-generated image similarity results, through radiologists and laypeople who can qualitatively evaluate which images they believe are the most realistic. This approach is often used by major GAN publications but requires considerable time and cost [24].

In Figure 4.7 the predicted images from a sample of highly occluded inputs are showcased. The results regarding *CycleGAN* are the same; the images are too blurry and bone structure significantly incorrect to be considered plausible X-rays. Between our approach and *pix2pix*, as seen in the second and third row, our approach produces better bone structure, avoiding fused and thin bone mass. Additionally, as seen in various fingers in all images, our approach produces a more distinguishable bone structure in the curled fingers, while *pix2pix* produces a more conservative indistinguishable output. Figure 4.10 enlarges these inconsistencies for clear visualization.

One clear conclusion from this set of experiments is the underwhelming results provided by the unsupervised *CycleGAN* approach. The benefit of using unsupervised learning is that images do not need to be labeled, specifically in our case that the input images and output X-rays do not need to be in pairs. This is a large benefit in medical research, where X-ray images can be found in abundance collected from past patients. The downside is that these images would not have the paired input, and thus can only be used in the unsupervised approach. Although these X-ray images can easily be found and used, it is clear that the unsupervised approach does not provide clear and accurate results. This is due to the lack of the *U-Net* used in *pix2pix*, which was derived from previous literature in improving biomedical image segmentation [54]. The *U-Net* architecture proves superior to the residual blocks used in *CycleGAN* due to the ability of *U-Net* to capture local features

in the down-sampling blocks and carry them over to the image being generated through the up-sampling blocks. Doing this generates an image with fine detail in various local regions, producing a clearer and less blurry result. As seen in Tmenova et al., *CycleGAN* is better used when transferring styles rather than generating structures [40]. By using *CycleGAN* to simply convert a synthetic X-ray to look like a more realistic X-ray, their work avoids the downside of the unsupervised approach while capitalizing on the benefit that they do not need to gather paired real X-rays.

Although our results show an improvement in the general approaches, it is evident that our work still has room for improvement. In all images showcased between Figure 4.6 and Figure 4.7, our results are not perfectly similar to the ground truth. This shows that there are still areas where our work must improve to create X-rays that are indistinguishable to the ground truth. Additionally, we compare our results to real X-rays taken from a bone phantom as seen in 4.8. We notice that although our predicted X-rays show plausible results, we still have a stark difference between a real X-ray. Much of this can be attributed to the simplified model we use for generating data. The difference between our results and real X-rays is a clear indicator that our future work must factor in the style of real X-rays, similar to the work of Tmenova et al., described previously.

A limitation of our current approach is the use of low-resolution input and output of 256 by 256. We chose this by design, as our initial goals were to establish baseline results for further research, and to simplify the computational requirements for shorter training times and faster iteration. Our future work will begin to explore producing X-rays of higher resolution. There is already a significant body of research in this area, such as *pix2pixHD*, which uses multiple inputs through a global generator and local enhancer to produce high-resolution outputs of 2048 by 1024 [47]. As seen in our review of literature, *Progressive GAN* architectures, which use multiple generators and discriminators to incrementally increase resolution are also a viable option, and have already been applied to mammography [3][7]. Our use of hand primitives has helped establish a solid baseline, but future work will need to incorporate more detailed models, textures, background, and hand shapes to establish a more realistic distribution of images. The results of this section establish the need for a task-specific approach to X-ray generation, and our next set of experiments will build by adding additional data to generate more accurate images.

4.3 3D Point Cloud Data

The evaluation of our final set of experiments using 3-D point cloud input data is displayed in Table 4.6. We compare the new results with the previously obtained similarity results from the RGB experiments, and to maintain similarity between experiments we use the same test dataset of 500 test images. Table 4.7 showcases the difference in results when the networks are tested on just occluded data, using the same 200 images occluded test dataset from the RGB experiments.

Qualitative results are shown in Figure 4.11, where the same sample inputs as used in Figure 4.6 are brought back so that comparison can be made between the new networks with 3-D input and the same networks without 3-D input. Further samples are showcased in Figure 4.12 which highlights occluded cases. A summary of the qualitative observations can be found in Table 4.8.

Table 4.6: Evaluation metrics of image similarity comparing *pix2pix* and our own *pix2xray* to their modified counterparts that include 3-D point cloud processing.

Network	Pixel Accuracy	SSIM	RMSE	PSNR
pix2pix	78.81	0.955	0.0552	25.38
pix2xray	78.27	0.967	0.0450	27.12
pix2pix + PointNet	79.93	0.969	0.0426	27.52
pix2xray + PointNet	82.93	0.970	0.0415	27.77

Table 4.7: Evaluation metrics of image similarity in unique test dataset with only occluded cases between the 3-D point cloud networks and the originals.

Network	Pixel Accuracy	SSIM	RMSE	PSNR
pix2pix	66.98	0.957	0.0579	24.97
pix2xray	82.24	0.967	0.0460	26.93
pix2pix + PointNet	79.34	0.969	0.0435	27.32
pix2xray + PointNet	82.02	0.970	0.0414	27.76

The quantitative results highlight the bump in image similarity driven by the point cloud input. When compared to the original networks, both *pix2pix* and our *pix2xray* modified with PointNet produce more accurate images. As seen in Table 4.6, *pix2pix* gains a 1 percent increase in pixel accuracy, 0.014 increase in SSIM, a 0.126 drop in RMSE, and 2.35 increase in PSNR. When looking at our *pix2xray* improvement, we see a much larger in pixel accuracy, and slight gains in SSIM of 0.003, a 0.0035 decrease in RMSE, and .65

Table 4.8: Summary of qualitative differences between outputs of networks using RGB and point cloud input together.

Network	Summary
<code>pix2pix</code>	Clear image but large inconsistencies in plausible bone structure and shape
<code>pix2xray</code>	Clear image with clear and plausible bone structure and shape
<code>pix2pix + PointNet</code>	Clear image but small inconsistencies in bone structure
<code>pix2xray + PointNet</code>	Highly clear image with clear and plausible bone structure and shape

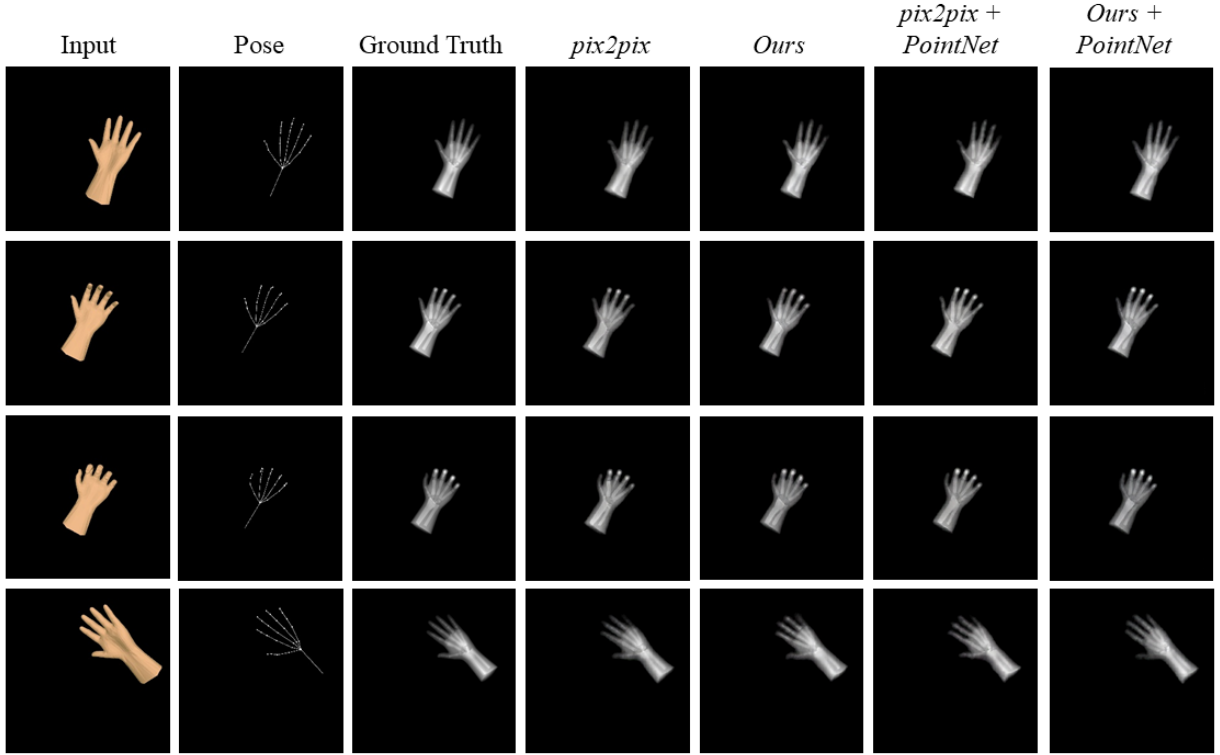


Figure 4.11: Comparison of predicted X-rays between *pix2pix*, *pix2xray*, and their modified counterparts that include *PointNet* modifications for 3-D point cloud input handling.

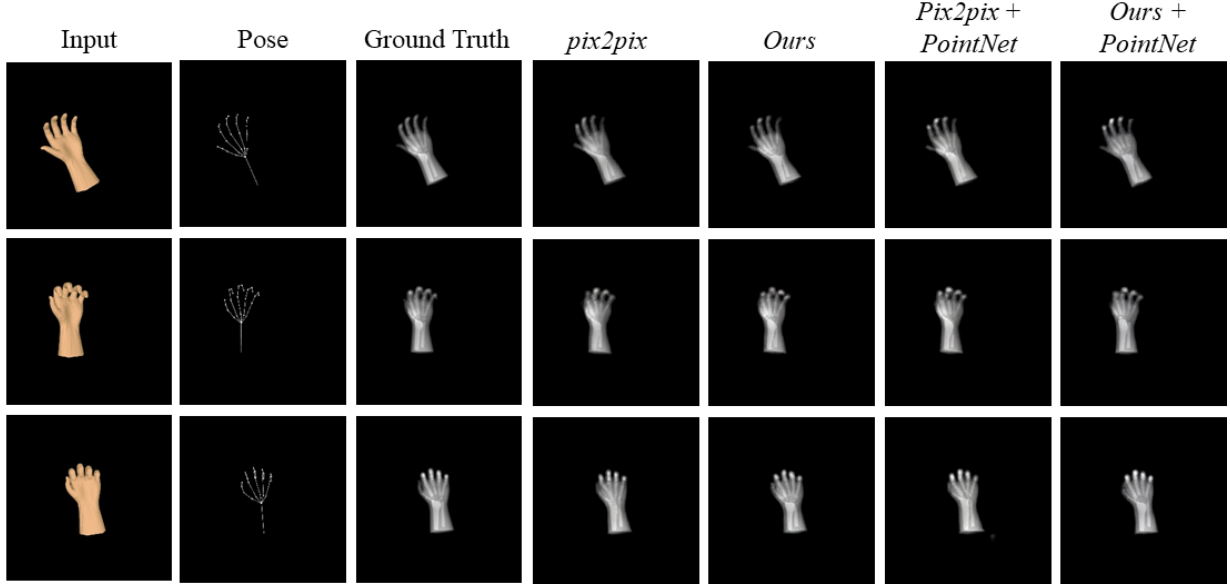


Figure 4.12: Comparison of predicted X-rays when handling occlusion between *pix2pix*, *pix2xray*, and their modified *PointNet* counterparts using occluded samples as input.

gain in [PSNR](#). The results are clear that our approach remains superior, displaying the highest image similarity across the board. In the previous section, we noticed that *pix2pix* maintained a higher pixel accuracy even while the other quantitative and qualitative results indicated our approach was superior. In this set of results, the addition of 3-D point cloud data has solved our low pixel accuracy issue, fixing the proposed problem of low luminance in our image from the additional black background input.

When analyzing the gain of each network when 3-D point cloud data is introduced, *pix2pix* gains the most in this modification, even though it still falls short of our approach. The reason for the disparity in gain is that our *pix2xray* approach already incorporates two pieces of information, the original image and information about the pose of the hand. Therefore, the addition of now the third piece of information into the scenario does not greatly benefit the network as the gain between both additional pieces overlaps. In comparison, *pix2pix* now gains the second piece of information for the first time, so we expect a gain in accuracy as the network now has two pieces of information to correctly generate structures in the X-ray compared to the simple original approach.

As done previously, we also analyze the outcomes of our experiments on a separate dataset of only occluded images. In this set of results showcased in [Table 4.7](#), we see again that *pix2pix* makes a gain in generating accurate X-rays, and our *pix2xray* approach makes very slight gains, getting better results only in [SSIM](#), [RMSE](#), and [PSNR](#), but dropping

slightly in pixel accuracy. From these results, we can conclude that 3-D point cloud data provides a benefit to handling occlusion cases as seen by the gain made by *pix2pix*. Our network is already designed to handle occlusion through the pose, and so the gains we make through adding point cloud input is smaller compared to *pix2pix*.

In Figure 4.11 and Figure 4.12, we compare the qualitative gains that 3-D point cloud input provides to our output images. In the previous section, we highlighted the fused bones in the third row and inconsistent bone structures of the fingers in various images produced by *pix2pix*. These inconsistencies are removed by the addition of 3-D point cloud data, as seen in the third row of Figure 4.11 through the fused bones. Figure 4.13 shows an enlarged version of the X-ray to highlight these differences. The benefits to our network become more visible in Figure 4.12, where we see the better bone structure of the metacarpal and finger bones in all rows. It is clear that the addition of 3-D point cloud data is of more importance to the more difficult cases rather than the more simple open hand poses, as the gains in Figure 4.12’s occluded samples are more visible than the changes in Figure 4.11’s simpler hand poses. Figure 4.14 shows an enlarged version from the occluded cases to clearly visualize these findings.

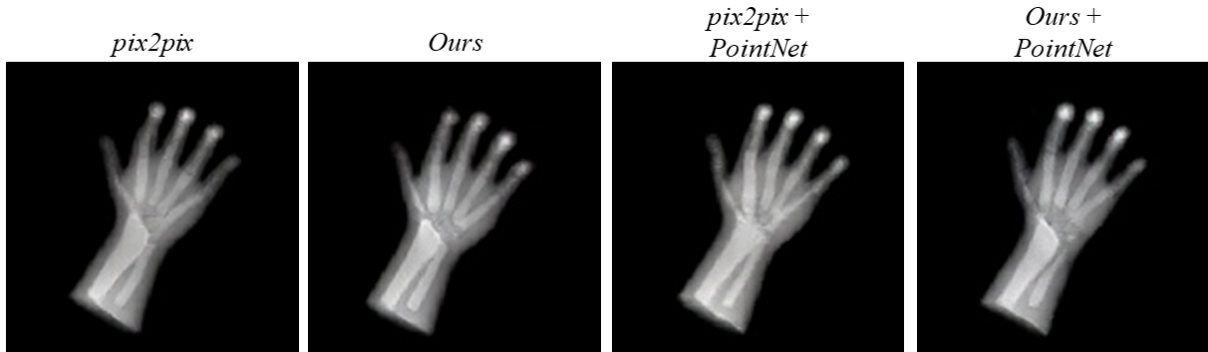


Figure 4.13: Enlargement of predicted X-rays generated from modified networks to visualize the inconsistencies.

In the end, the addition of 3-D point cloud data provides an additional avenue for improvement to the baseline approaches, but there are limitations to incorporating 3-D point cloud data. Firstly, the inclusion of the branched network facilitating 3-D point cloud data processing through *PointNet* induces a significant computational increase in the network. In our experiments, training time increased three-fold, and the time to make a single prediction also approximately doubled. These results can be clearly seen in Table 4.9, which highlights the number of parameters and the time taken to complete 10 predictions for each network based on our available hardware. Where before a 256x256

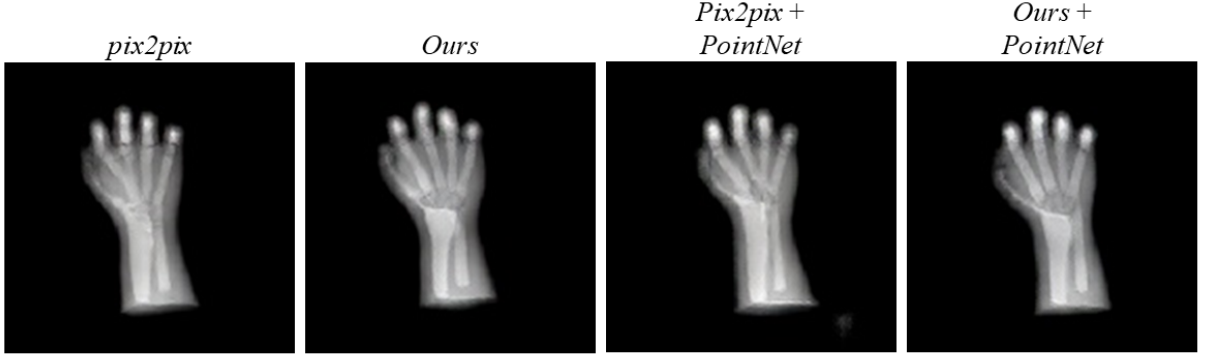


Figure 4.14: Enlargement of predicted X-rays generated from modified networks using samples from our occluded set to visualize the inconsistencies.

image could be generated within a second using the GPU, the same image takes around four seconds in the new networks. This increase in computational complexity will influence the applicability of the work. As stated previously, the possible applications of this work might be incorporation into AR technology, or in mobile technology. The increased computational power needed would inhibit it from being accessible to these applications.

Table 4.9: Comparison of network complexity between networks based on number of parameters and time taken to test 10 images in seconds.

Network	Number of Parameters	Time in Seconds
pix2pix	54,425,859	0.738
CycleGAN	6,962,369	0.544
pix2xray	108,857,862	1.341
pix2pix + PointNet	61,374,796	2.012
pix2xray + PointNet	115,802,727	2.531

The second limitation of our current approach is the lack of handling point cloud variability. Point clouds density and shape are highly variable. A typical depth camera will only produce a partial point cloud, and the density is variable on distance, perspective, and camera technology. In the previous sections, we highlighted a need to use a preprocessing network to handle different types of point cloud image inputs, and create an intermediate representation for the main generator. In implementing *PointNet*, this type of approach is not possible, as *PointNet* examines the unordered point set for spatial relations. Thus, using point clouds of lower density and limited shape would result in less information being drawn. Therefore, although 3-D point cloud data provides benefits to generating X-rays, it does come with limitations that need to be addressed moving forward.

Chapter 5

Conclusion

5.1 Summary

Synthetic X-rays are restricted to applications where 3-D models or [CT](#) scans are available. The purpose of this thesis is to explore the feasibility of using atypical input modalities to estimate an X-ray image. To accomplish this, we created a custom generator application that uses an X-ray simulation environment to generate images. Our generator is capable of generating large datasets of images, including point cloud, [RGB](#), and pose images. Using these datasets, we trained multiple [GANs](#) to accomplish image translation between these inputs and X-ray images. Our first set of experiments examined point cloud images on the *pix2pix* architecture and proved that image translation between point clouds and X-rays is possible. We further examined the difference in point cloud densities and their effect on generating clearer X-rays. Our results show that higher densities produce clearer results, but images that had occlusion failed to generate plausible X-rays.

To solve this, we tested a novel architecture named *pix2xray* that incorporated hand pose into estimating X-rays. Supervised learning through paired images performs better than unsupervised learning. In comparison to the *pix2pix* and *CycleGAN* networks, our novel architecture performed better especially in occluded cases. The inclusion of the pose as a secondary input provided tangible benefits to the generated image. The novel architecture produced clearer bone structure, positioning the bones without errors to create plausible X-rays.

We then tested the effect of 3-D point cloud input into our network. We modified both *pix2pix* and our *pix2xray* to include a branch that processes 3-D point cloud data. Using the modified networks, we saw gains in image similarity where the point cloud data helped

to produce clearer bone structures. Our modified *pix2xray* architecture still maintained superiority over the modified *pix2pix* and the original networks. Our work proves that generating X-ray images using inputs such as RGB images and point clouds is possible, and produces plausible and visually pleasing results that can be used for non-diagnostic visualization purposes.

5.2 Objectives

The following objectives were fulfilled through the methods and results of this thesis:

Objective 1: Development of a custom dataset generator application capable of generating X-ray images, RGB images, pose images, and point cloud images in a flexible and fast manner.

A custom pipeline was developed using a custom simulation environment made in C++, capable of generating a dataset of images including X-rays, poses, and point clouds. The custom generator provided large datasets for network training and remains flexible for future improvements.

Objective 2: Evaluation of point cloud image to X-ray image translation, and the effect of point cloud density on the outcome.

pix2pix was implemented in evaluating point cloud image translation. Higher point cloud densities proved significantly better in producing more realistic predicted X-rays, but all point cloud densities showed the ability to generate X-rays to some degree.

Objective 3: Development of novel architecture extended from *pix2pix* and evaluation of a novel approach to original for RGB image to X-ray image translation.

A novel architecture named *pix2xray* was developed, which extends the *pix2pix* architecture by incorporating the hand pose. The novel architecture performed significantly better and proved superior to both *pix2pix* and *CycleGAN*. the unsupervised *CycleGAN* approach proved highly ineffective at producing clear X-rays compared to *pix2pix* and our *pix2xray*. Our *pix2xray* approach handled occluded cases better than the other approaches, producing accurate and well-defined bone structures.

Objective 4: Evaluation of integrating 3-D point cloud data into novel architecture and *pix2pix*.

pix2pix and our own *pix2xray* were modified to include a branch that processes 3-D point cloud data and integrates it into generating the final X-ray image. The modified

pix2pix gained significant image similarity improvements, while our *pix2xray* gained a slight edge. *pix2xray* with the 3-D point cloud input provided the best results of any previous approach.

5.3 Future Work

Throughout our work, we have established a strong baseline for image to X-ray translation. We have identified multiple areas for future work that include improvement and applications of these initial findings.

5.3.1 Data Variation

The major limitation in our custom dataset pipeline was the use of a simple primitive hand model. As we noted, our synthetic X-rays are still significantly different from a real X-ray look. To create a more realistic dataset, with a greater distribution of anatomy types, future work will need to incorporate realistic hand models. This includes hand models that have significant soft tissue definition, with muscles, tendons, and other structures defined. This will result in synthetic X-rays that more closely resemble their real counterparts. Furthermore, our use of a single hand size, color, and shape, causes our networks to become fit on that specific size. Future work will need to incorporate different hand sizes, skin colors, and textures due to aging, to create a dataset more representative of the general population.

5.3.2 Anatomy

In this thesis, we focused on the anatomy of the hand, due to our original motivation of translating images of a surgeon’s hands into X-rays and merging them with patient X-rays in the [CAM-C](#) system. Future work will be directed towards other anatomical locations, producing X-rays of areas such as the chest and abdomen. Moving towards these areas will require reconsideration of our *pix2xray* architecture, as many other areas of the body lack would have a single pose line, but multiple major organs and structures to be generated. Instead of generating pose images, in this scenario, it could be more beneficial to generate an organ map. Similar to a pose image, an organ map might define the general location and shape of major structures from the original input and could be used to generate a better result as is done in *pix2xray*.

5.3.3 Partial Point Clouds

Our use of whole point clouds assumes that end-users will have a complete point cloud of the anatomy available. This is not always the case, the completeness of the point cloud is dependant on the technology being used. A future direction in this area would be a way to eliminate the variability in point clouds. Similar to the pose image, or the organ map discussed above, point clouds could be processed in a preliminary network that produces this intermediate data type, which can then be used to generate the images. We have already proved that this approach works with [RGB](#) images and poses, and so partial point clouds and differences in density could be overcome through this method.

5.3.4 User Validation

Throughout this thesis, we evaluated the various networks using similarity metrics found in the literature. It is evident that these metrics can often fluctuate, and could mislead the findings. Examining the qualitative results by eye provides more meaning and validation, as small changes in quantitative values are meaningless without seeing their visual outcome. To this end, as this work progresses towards more realistic X-rays and specific applications, it will be necessary to move towards a user study protocol. Using radiologists, students, or even lay people to visually validate results would provide metrics that are more meaningful and easier to interpret. Our end goal is to make these synthetic X-rays visually plausible to the end-user, and so using the end-user to validate the image is the most optimal choice. User validation was out of reach for this thesis, but future work will incorporate these studies to provide a meaningful evaluation.

With this baseline established, we can look towards improving our approach and refining and methods, in the hopes that our work can easily replace state-of-the-art approaches more flexibly. We look forward to applying the principles of this research to specific applications in [AR](#), surgical simulations, and other biomedical visualizations. The ability to generate realistic X-rays from any input holds countless possibilities and will revolutionize how we utilize synthetic X-rays in the future.

References

- [1] U. Bottigli, A. Brunetti, B. Golosio, P. Oliva, S. Stumbo, L. Vincze, P. Randaccio, P. Bleuet, A. Simionovici, and A. Somogyi. Voxel-based Monte Carlo simulation of X-ray imaging and spectroscopy experiments. *Spectrochimica Acta Part B: Atomic Spectroscopy*, 59(10):1747–1754, October 2004.
- [2] Franck P. Vidal, Manuel Garnier, Nicolas Freud, Jean Michel Létang, and Nigel W. John. *Simulation of X-Ray Attenuation on the GPU*. The Eurographics Association, 2009.
- [3] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. *arXiv:1710.10196 [cs, stat]*, February 2018.
- [4] Scott Reed, Zeynep Akata, Xinchun Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative Adversarial Text to Image Synthesis. *arXiv:1605.05396 [cs]*, June 2016.
- [5] Nassir Navab, Sandro-Michael Heining, and Joerg Traub. Camera Augmented Mobile C-Arm (CAMC): Calibration, Accuracy Study, and Clinical Applications. *IEEE Transactions on Medical Imaging*, 29(7):1412–1423, July 2010.
- [6] Osama M. Dorgham, Stephen D. Laycock, and Mark H. Fisher. GPU Accelerated Generation of Digitally Reconstructed Radiographs for 2-D/3-D Image Registration. *IEEE Transactions on Biomedical Engineering*, 59(9):2594–2603, September 2012.
- [7] Dimitrios Korkinof, Tobias Rijken, Michael O’Neill, Joseph Yearsley, Hugh Harvey, and Ben Glocker. High-Resolution Mammogram Synthesis using Progressive Generative Adversarial Networks. *arXiv:1807.03401 [cs]*, September 2019.

- [8] Xingyi Yang, Nandiraju Gireesh, Eric Xing, and Pengtao Xie. XRayGAN: Consistency-preserving Generation of X-ray Images from Radiology Reports. *arXiv:2006.10552 [cs, eess]*, June 2020.
- [9] W. C. Röntgen. On a New Kind of Rays. *Science*, 3(59):227–231, February 1896.
- [10] M. H. H. Noordeen, N. Shergill, R. S. Twyman, J. P. Cobb, and T. Briggs. Hazard of ionizing radiation to trauma surgeons: Reducing the risk. *Injury*, 24(8):562–564, September 1993.
- [11] Germanas Marinskis, Maria Grazia Bongiorno, Nikolaos Dagres, Thorsten Lewalter, Laurent Pison, and Carina Blomstrom-Lundqvist. X-ray exposure hazards for physicians performing ablation procedures and device implantation: Results of the European Heart Rhythm Association survey. *EP Europace*, 15(3):444–446, March 2013.
- [12] Tarun Dusad, Vishal Kundnani, Shumayou Dutta, Ankit Patel, Gaurav Mehta, and Mahendra Singh. Comparative Prospective Study Reporting Intraoperative Parameters, Pedicle Screw Perforation, and Radiation Exposure in Navigation-Guided versus Non-navigated Fluoroscopy-Assisted Minimal Invasive Transforaminal Lumbar Interbody Fusion. *Asian Spine Journal*, 12(2):309–316, April 2018.
- [13] Ronald E. Vlietstra and Louis K. Wagner. X-ray Burns—Painful, Protracted, and Preventable. *Clinical Cardiology*, 31(4):145–147, April 2008.
- [14] B. De Man, J. Nuyts, P. Dupont, G. Marchal, and P. Suetens. Metal streak artifacts in X-ray computed tomography: A simulation study. In *1998 IEEE Nuclear Science Symposium Conference Record. 1998 IEEE Nuclear Science Symposium and Medical Imaging Conference (Cat. No.98CH36255)*, volume 3, pages 1860–1865 vol.3, November 1998.
- [15] M. Tapiovaara, M. Lakkisto, and A. Servomaa. PCXMC. A PC-based Monte Carlo program for calculating patient doses in medical x-ray examinations. Technical Report STUK-A-139, Finnish Centre for Radiation and Nuclear Safety (STUK), 1997.
- [16] M. R. Ay, M. Shahriari, S. Sarkar, M. Adib, and H. Zaidi. Monte Carlo simulation of x-ray spectra in diagnostic radiology and mammography using MCNP4C. *Physics in Medicine and Biology*, 49(21):4897–4917, October 2004.
- [17] N. Freud, P. Duvauchelle, J. M. Létang, and D. Babot. Fast and robust ray casting algorithms for virtual X-ray imaging. *Nuclear Instruments and Methods in Physics*

Research Section B: Beam Interactions with Materials and Atoms, 248(1):175–180, July 2006.

- [18] Franck P. Vidal and Pierre-Frédéric Villard. Development and validation of real-time simulation of X-ray imaging with respiratory motion. *Computerized Medical Imaging and Graphics*, 49:1–15, April 2016.
- [19] D. Kundur and D. Hatzinakos. Blind image deconvolution. *IEEE Signal Processing Magazine*, 13(3):43–64, May 1996.
- [20] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T. Roweis, and William T. Freeman. Removing camera shake from a single photograph. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 787–794, New York, NY, USA, July 2006. Association for Computing Machinery.
- [21] Yichang Shih, Sylvain Paris, Frédo Durand, and William T. Freeman. Data-driven hallucination of different times of day from a single outdoor photo. *ACM Transactions on Graphics*, 32(6):1–11, November 2013.
- [22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *arXiv:1406.2661 [cs, stat]*, June 2014.
- [23] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *arXiv:1411.1784 [cs, stat]*, November 2014.
- [24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, Honolulu, HI, July 2017. IEEE.
- [25] Ali Madani, Mehdi Moradi, Alexandros Karargyris, and Tanveer Syeda-Mahmood. Semi-supervised learning with generative adversarial networks for chest X-ray classification with ability of data domain adaptation. In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pages 1038–1042, April 2018.
- [26] Xingde Ying, Heng Guo, Kai Ma, Jian Wu, Zhengxin Weng, and Yefeng Zheng. X2CT-GAN: Reconstructing CT from Biplanar X-Rays with Generative Adversarial Networks. *arXiv:1905.06902 [cs, eess]*, May 2019.

- [27] Yuewen Sun, Ximing Liu, Peng Cong, Litao Li, and Zhongwei Zhao. Digital radiography image denoising using a generative adversarial network. *Journal of X-Ray Science and Technology*, 26(4):523–534, 2018.
- [28] L. Lemieux, R. Jagoe, D. R. Fish, N. D. Kitchen, and D. G. T. Thomas. A patient-to-computed-tomography image registration method based on digitally reconstructed radiographs. *Medical Physics*, 21(11):1749–1760, 1994.
- [29] Maryam Babaei and Ahmad Reza Naghsh Nilchi. Synthetic data generation for X-ray imaging. In *2014 21th Iranian Conference on Biomedical Engineering (ICBME)*, pages 190–194, November 2014.
- [30] George W. Sherouse, Kevin Novins, and Edward L. Chaney. Computation of digitally reconstructed radiographs for use in radiotherapy treatment design. *International Journal of Radiation Oncology*Biophysics*, 18(3):651–658, March 1990.
- [31] Daniel Ruijters, Bart M. ter Haar-Romeny, and Paul Suetens. GPU-accelerated digitally reconstructed radiographs. In *Proceedings of the Sixth IASTED International Conference on Biomedical Engineering*, BioMED ’08, pages 431–435, USA, February 2008. ACTA Press.
- [32] D.B. Russakoff, T. Rohlfing, K. Mori, D. Rueckert, A. Ho, J.R. Adler, and C.R. Maurer. Fast generation of digitally reconstructed radiographs using attenuation fields with application to 2D-3D image registration. *IEEE Transactions on Medical Imaging*, 24(11):1441–1454, November 2005.
- [33] Tatsuki Koga, Naoki Nonaka, Jun Sakuma, and Jun Seita. General-to-Detailed GAN for Infrequent Class Medical Images. *arXiv:1812.01690 [cs, stat]*, November 2018.
- [34] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. *Neurocomputing*, 321:321–331, December 2018.
- [35] Elman Mansimov, Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Generating Images from Captions with Attention. *arXiv:1511.02793 [cs]*, February 2016.
- [36] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5908–5916, Venice, October 2017. IEEE.

- [37] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks. *arXiv:1711.10485 [cs]*, November 2017.
- [38] Brian Teixeira, Vivek Singh, Terrence Chen, Kai Ma, Birgi Tamersoy, Yifan Wu, Elena Balashova, and Dorin Comaniciu. Generating Synthetic X-Ray Images of a Person from the Surface Geometry. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9059–9067, Salt Lake City, UT, June 2018. IEEE.
- [39] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved Training of Wasserstein GANs. *arXiv:1704.00028 [cs, stat]*, December 2017.
- [40] Oleksandra Tmenova, Rémi Martin, and Luc Duong. CycleGAN for style transfer in X-ray angiography. *International Journal of Computer Assisted Radiology and Surgery*, 14(10):1785–1794, October 2019.
- [41] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *arXiv:1703.10593 [cs]*, November 2018.
- [42] Blender Foundation. Blender.org - Home of the Blender project - Free and Open 3D Creation Software.
- [43] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks. In *ACM Transactions on Graphics*, volume 33, August 2014.
- [44] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: An Open-Source Mesh Processing Tool. *Eurographics Italian Chapter Conference*, page 8 pages, 2008.
- [45] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked Hourglass Networks for Human Pose Estimation. *arXiv:1603.06937 [cs]*, July 2016.
- [46] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015.
- [47] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8798–8807, Salt Lake City, UT, USA, June 2018. IEEE.

- [48] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. *arXiv:1711.06396 [cs]*, November 2017.
- [49] Stefan Milz, Martin Simon, Kai Fischer, and Maximillian Pöpperl. Points2Pix: 3D Point-Cloud to Image Translation using conditional Generative Adversarial Networks. *arXiv:1901.09280 [cs]*, September 2019.
- [50] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *arXiv:1612.00593 [cs]*, April 2017.
- [51] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.
- [52] Keyang Cheng, Rabia Tahir, Lubamba Kasangu Eric, and Maozhen Li. An analysis of generative adversarial networks and variants for image synthesis on MNIST dataset. *Multimedia Tools and Applications*, 79(19):13725–13752, May 2020.
- [53] Kiana Ehsani, Roozbeh Mottaghi, and Ali Farhadi. SeGAN: Segmenting and Generating the Invisible. *arXiv:1703.10239 [cs]*, May 2018.
- [54] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv:1505.04597 [cs]*, May 2015.