Princeton Advanced Computer Graphics (COS 526) Final Project: First Steps Towards Novel View Synthesis on Sketches

Maxine Perroni-Scharf

Abstract

Novel view synthesis is a well-researched problem in computer vision and computer graphics. View synthesis aims to generate novel views from one or more given source views. This work takes the first steps towards novel view synthesis on black and white sketch style line-drawings, which have no meaningful 3D representations. We develop a system that takes in sparse frame line drawings of the same object from different poses, and animates between these frames. This is achieved through an encoder, latent-space interpolation, and decoder approach. We perform various transforms on our training dataset, which prove to be key to the success of this approach, and compare the performance of our approach using both a GAN and a VAE.

1. Introduction

Sketches and animation are widely used as an art form, and have various applications in computer vision and computer graphics. Traditionally, 2D hand-drawn animation is achieved by drawing frame-by-frame still images [20]. In old cartoons, scenes would usually take place from a single viewpoint, because of the extra complexities that arise from animating a 3D consistent moving background with changing perspectives.

This work aims to overcome this problem by taking a step towards a system where an artist could input sparse hand-drawn frames of a scene, and automatically interpolate between these frames to create moving camera-perspective animations. We believe that novel view synthesis could be used for this purpose, and while novel view synthesis and automatic animation techniques already exist for natural and synthetic images, there is currently no existing approach for novel view synthesis on sketches.

Black-and-white line sketches are more ambiguous than natural images, and lack color, shadows, and light which can all be utilized to infer the structure of a subject. The challenge in developing this system is that we need to learn about the three-dimensional behaviour of apparently two-dimensional input images. To overcome this challenge, we explore a simple baseline approach without attempting to explicitly find 3D representations of our subjects. We do this by using generative models to encode and decode sketch frames, and interpolate between them in latent space.

2. Related Work

Novel View Synthesis Novel view synthesis is a long standing task in computer vision. Seminal work in novel view synthesis interpolates between input images directly, using pixel-wise or optical-flow based methods [4]. Predominantly in recent approaches to novel view synthesis, a model is trained to create some kind of 3D representation such as a mesh, point-cloud, voxel-grid or neural radiance field [18, 12, 13, 19], which can then be rendered from different angles.

The use of auto-encoders and latent-space interpolation for novel view synthesis has also been explored extensively. Supervised approaches include using pose information to disentangle light, shape, identity and and pose from the latent codes of encoded images [9]. These approaches prove to be challenging and suffer from view inconsistencies. For natural images, the 3D-representation-based approaches discussed above are typically more successful.

Sketches Line drawing cartoon scenes introduce new challenges to the problem of novel view synthesis. Typically drawn images of the same subject contain structural inconsistencies at different angles. Furthermore, grayscale line drawings do not encode as much information as photographs, and cannot be easily used to train a NeRF or to generate a 3D representation of an object. Therefore, the task of novel-view synthesis for sketches has not been widely explored.

In the past, an image segmentation approach [5] has been developed to morph between different views of cartoon images. Wire art 3D reconstruction has also been achieved by matching candidate curve segments to several multi-view input images of wire sculptures [11]. However, these methods rely on the input images to be formed from a series of looped lines, and are specifically tailored to cartoon faces and smooth pieces of wire respectively. This project instead aims to develop a more general approach to this task by

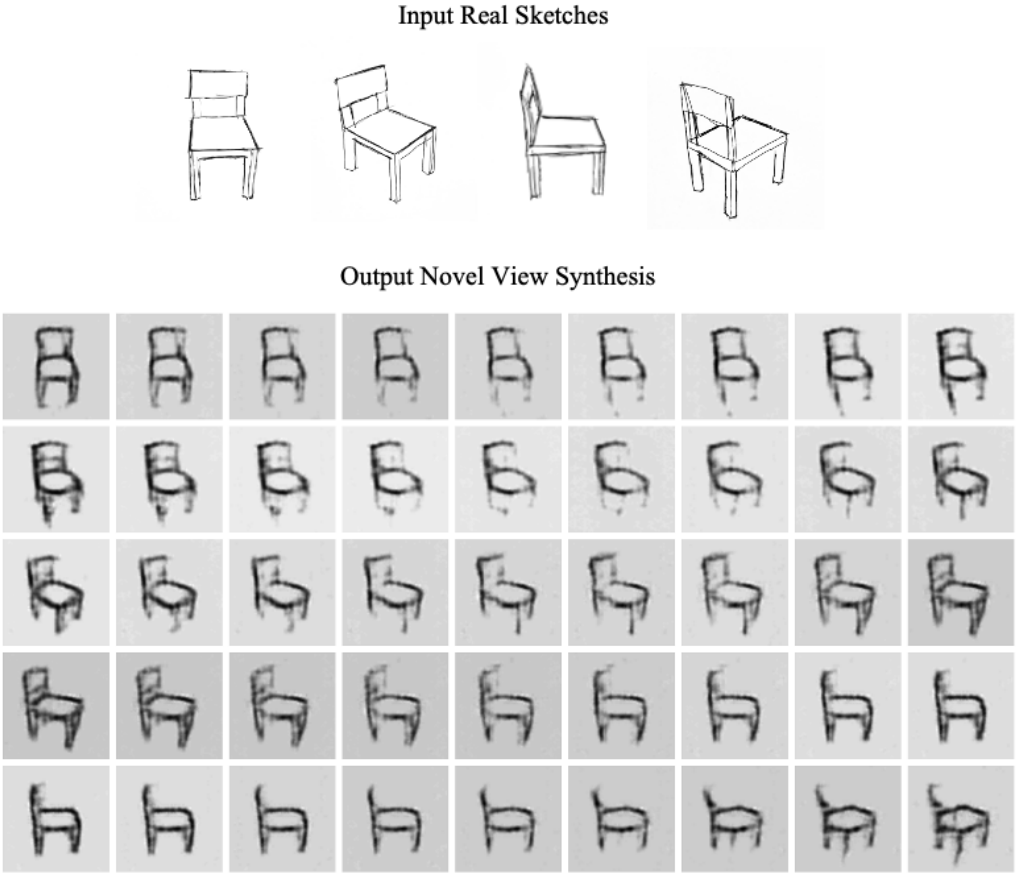

Figure 1. Input real sketches (top) and the results from our approach (bottom). In this example, each input frame is interpolated with 15 between frames. If you view the cells left-to-right, row-by-row, the chair appears to rotate counter-clockwise.

using known GAN-based methods that can be trained on different datasets.

Other works also explore the applications of AI for animation. AnimeGAN combines neural style transfer and GANs to transform photorealistic animations into colored anime-style animations [3]. More recently, pose estimation has been used to rig and animate children's drawings of figures [1].

SketchGAN [10] uses a cascade encoder-decoder network for sketch completion and classification. It focuses on the task of dealing with incomplete, ambiguous black-on-white sketch data from the Sketchy database [17].

View-dependent geometry [14] tackles the problem of dealing with view inconsistencies that arise from hand-drawn figures. In this work, a prior 3D input model is deformed based on reference drawings of a subject viewed from different angles. This work requires strong prior information as it relies on a hand-crafted 3D model for each new subject, and therefore would defeat the purpose of frame interpolation as a means of saving an animator time.

Generative Models Generative models play a key role in novel view synthesis related tasks. Generative models are used to create new data that is in line with a distribution of training data. In this work, I use a DCGAN [15], which is the classic GAN-based approach for generating images. I also use a previously proposed GAN inversion technique to map from images to latent space [6]. Variational auto-encoders can also be applied to the task of novel view synthesis, and I use the approach outlined in the original VAE paper in this project [8].

3. Theory

To achieve novel view synthesis on sketches, we used an encoder-decoder approach as can be seen in figure 2. Each input frame is encoded into a latent code. Then, we linearly interpolate between these latent codes to generate a new set of latent codes for each pair of input frames. Then these latent codes, including the codes for the two input images, are turned back into images using a decoder. The final an-

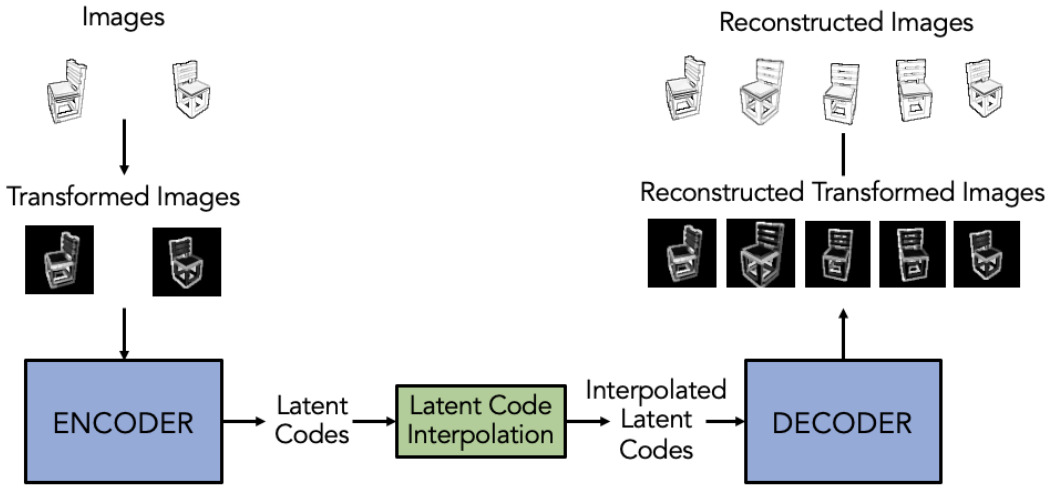

Figure 2. The pipeline we used for this project. There is an encoder, latent space interpolation and a decoder.

imation result is the sequence of images produced by this method.

To achieve the task of encoding and decoding images, we attempted two different methods. The first method was to use a variational auto-encoder (VAE) [8], which is trained end-to-toe with the aim of minimising the reconstruction loss of input images. The second approach was to use a generative adversarial network (GAN) [15] to learn to generate images from random latent codes, and a GAN inverter to map input images to latent codes based on the trained GAN.

3.1. Dataset Creation

To train our models, we required a dataset of black-on-white line drawings. To create this dataset, we used rendered images of the shapenet chair class, where each chair is captured at 26 random view-angles. Then, for each render, we took the grayscale version of the render, and the dilated grayscale version of the render, and calculated the pixel-wise difference of these two images. This results in a black-on-white contour image of each shapenet render as can be seen in figure 3.

3.2. Encoder/Decoder Architecture

VAE The first approach to encoding and decoding images is a variational auto-encoder. We use the classic VAE architecture [8] as outlined in the following paragraph.

The VAE is made up of an encoder and a decoder. The encoder consists of two 2D convolutions with ReLUs, and two linear layers to map input images of size 28x28 to an estimated mean and standard deviation of the distribution of 2-dimensional latent codes corresponding to that input image. In our version of the VAE, each convolutional layer has kernel size of 3, stride of 1 and padding of 1 (in the original VAE paper, a kernel size of 4 is used [8]). This is followed by sampling a latent vector from this distribution and using the re-parametrization trick to allow back-propagation through the sampling stage.

The decoder consists of a fully connected layer, a transpose 2D convolution, a ReLU, another transpose 2D convolution and finally a sigmoid layer, which maps input latent vectors back to reconstructed input images. Each tranpose 2D convolutional layer has kernel size of 3, stride of 1 and padding of 1 (in the original VAE paper, a kernel size of 4 is used [8]). Our VAE architecture has a capacity of 64, as per the original VAE paper [8].

The entire process is trained on the chair dataset, and we minimise the pixel-wise reconstruction loss as well as the KL loss for the latent-code distributions.

GAN The second approach to encoding and decoding images is a GAN followed by GAN Inversion. We use a classic DCGAN architecture [15]. The GAN itself acts as a decoder in this case and comprises of a generator G and a discriminator D.

The generator G performs five 2D transpose convolutions on an input latent code to produce a resulting image of size 64x64. Each convolutional layer has a kernel size of 4, stride of 1 and padding of 0, and is followed by a batch normalization layer and ReLU layer. The discriminator performs five 2D convolutions on a given image to map it to a single value indicating the discriminator's belief if a sample is fake or real. Each convolutional layer uses a kernel size of 4, stride of 2 and padding of 1. We minimize the generator loss and discriminator loss to train the GAN to generate images that trick the discriminator into believing the images

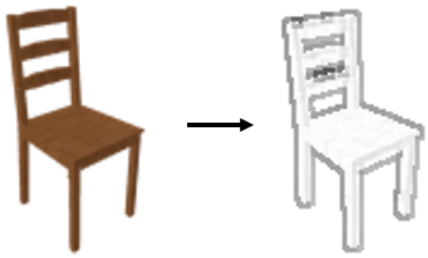

Figure 3. Rendered chair from Shapenet (left), and the results of turning this render into a contour sketch (right).

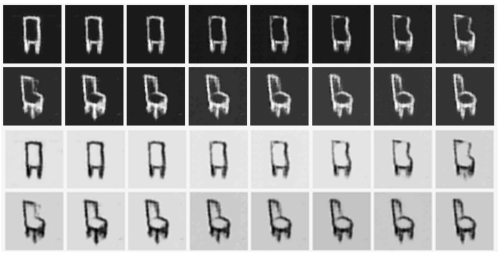

Figure 4. Example of outputs from our GAN (top), and sketchified results (bottom) achieved by erosion and black-white inversion.

belong to same distribution as the real training data [15].

GAN Inversion GAN inversion acts as a way to encode input image frames to latent codes. We use a simple method outlined in the GAN Inversion paper [6]. For each input image, we want to infer a latent code z such that G(z) produces an image that is as close as possible to the input image. Therefore, the training loop for GAN inversion consists of randomly sampling a z, and updating it based on the MSE loss between an input image and the image produced by G(z).

Re-Sketchifying our Images After we retrieve between-frames using our method, we turn these back into black and white sketches via image erosion and dilation. The results of this are seen in figure 4.

Implementation I implemented this project in python, using pytorch. I used the DCGAN code from the class assignment, and a vanilla VAE implementation from Google Colab [2] as starting points for the project.

4. Analysis

For both the VAE and GAN approaches, training on the sketch data directly did not work and produced bad results. For the VAE, no meaningful reconstruction occured, and for the GAN, we suffered from the discriminator loss jumping

Figure 5. The results of training a DCGAN on the sketch data directly. All eight result samples look the same, and bear no resemblance to the training data.

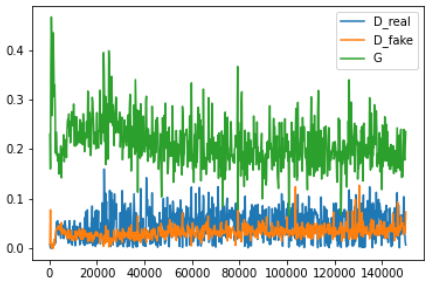

Figure 6. Discriminator and generator losses over time. The discriminator real loss is blue, the discriminator fake loss is orange and the generator loss is green.

to 0 and all outputs looking the same and not resembling chairs but rather striped noise as can be seen in figure 5.

To solve these problems we took several steps. Firstly, we pre-processed our input images to invert and dilate them. Image inversion was motivated by experimenting on the inverted MNIST digits dataset, with black digits on white backgrounds, and finding that the GAN training was significantly less stable with black on white than white on black.

Image dilation was motivated by the fact that the kernel size in our architecture was larger than the width of many of the thin lines in our sketches, and that these convolutional layers may have been distorting our input data. Image dilation is achieved by convolving a kernel with an input image. Each pixel is converted to white if at least one pixel under the kernel at that point is white. This results in the white contour of the subject thickening, making it more resistant to the effects of the convolutional layers in the GAN and VAE models.

Additionally, we experimented with many different parameter settings for our models. Ultimately, we reduced the kernel size in all of the convolutional layers in our VAE and GAN [8] [15]. For the GAN, we also used gradient penalty, updated the generator 2 times per discriminator update, and used Adam's optimizer with a learning rate of 0.00003. This

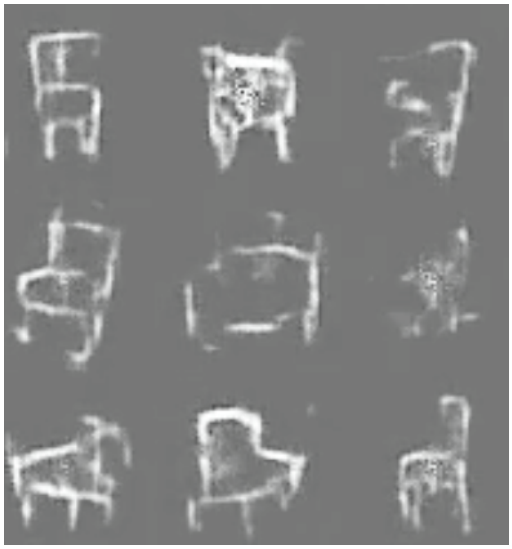

Figure 7. The results of our GAN after 10 epochs. These generated samples somewhat resemble chairs, but have a lot of structure problems and artifacts.

significantly improved the balance of our GAN training, as can be seen in figure 6. The combination of these elements allowed our GAN to achieve much more meaningful, although still not nearly perfect outputs. These can be seen in figure 7.

5. Results

We present qualitative results from testing our pipeline on a series of different input frames, including real sketches as can be seen in figure 1. We trained for on 22,000 chair sketch images (we used a subset of the chair class renders due to time limitations). The results and parameter settings that were used are outlined below.

GAN We trained our GAN for 20 epochs, using a batch size of 4. To test different inputs to the system, we took 7 sketch frames that were evenly distributed out over a full 360 degree rotation of each given test chair. We then trained the GAN inverter on these test frames, for 10,000 epochs. We interpolated to get 10 new views between input frames. We tested this on several different input chairs, and a representative sample of the results from this method can be seen in figure 8. We also tested our model on real sketches, which produced promising results as can be seen in figure 1.

The interpolation achieves some form of 3D understanding. For each test case, the apparent geometry of the object remains consistent (e.g. a couch remains a couch, and a round chair remains a round chair). Several pieces of information are lost during rotation, and there are many gaps

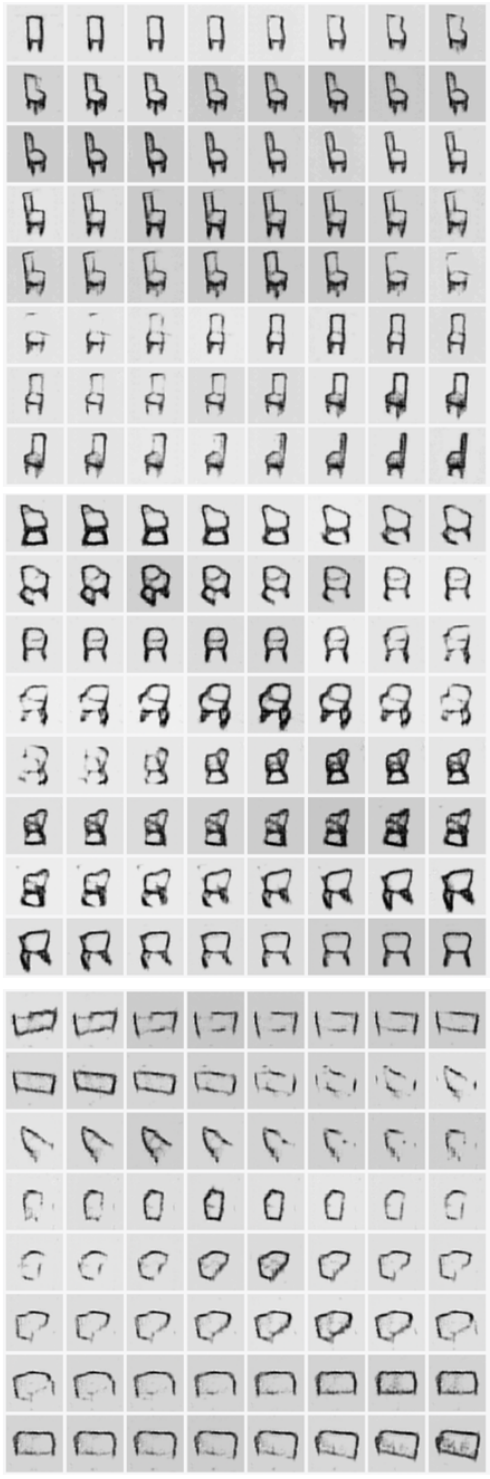

Figure 8. Novel view synthesis using the GAN approach on 3 different test cases: a dining room chair, a lounger and a couch. If you view the cells left-to-right, row-by-row, the chairs appear to rotate.

Figure 9. Novel view synthesis using the VAE approach on a couch. While reconstruction is good, if you view the cells left-to-right, row-by-row, the couch does not continue to turn in the same direction throughout interpolation, and several frames are simply blended together.

Figure 10. Example of the frame-blending problem from the VAE approach.

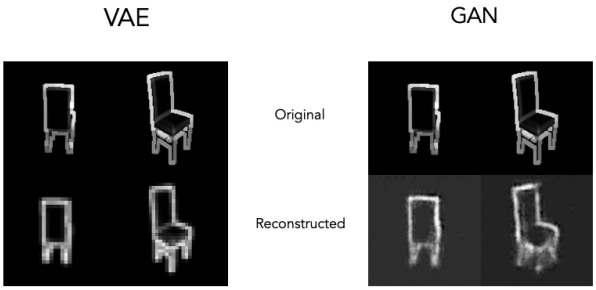

Figure 11. Reconstructions of input chair frames (top row) using the VAE approach (left) and the GAN approach (right).

introduced into the outlines of the chairs particularly when the object is being viewed from the back. The images also suffer from a mixture of artifacts caused by poor reconstruction.

VAE For the VAE approach we also took 7 sketch frames that were evenly distributed out over a full 360 degree rotation of each given test chair. We then trained the VAE on these test frames for 500 epochs. We interpolated to get 10 new views between input frames. The results from this method can be seen in fig 9.

The VAE reconstruction is much better than the GAN reconstruction as is seen in figure 11. Unlike the GAN approach, the VAE is trained end-to-toe with minimizing the reconstruction loss of the input frames in mind (whereas only the GAN inversion portion minimizes this reconstruction loss). However, the VAE is unable to meaningfully rotate the object with latent space interpolation. The rotations often ”swing back” around in the opposite direction than desired (e.g. if a two chair frames are 30 degrees apart, interpolating between their latent VAE code might apparently rotate backwards through 330 degrees rather than through 30 degrees). Furthermore, the VAE suffers from blending images together, rather than rotating them. This is highlighted in figure 10.

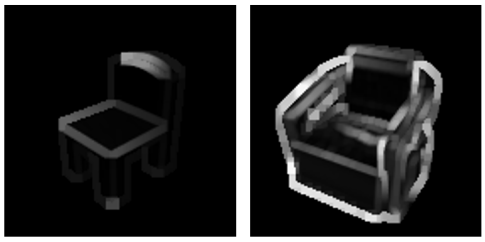

Figure 12. Examples of bad contour retrieval and inversion. These training images significantly reduce the quality of our dataset.

6. Discussion

The results from our GAN approach show meaningful rotations on real sketch data can be achieved. The VAE approach, however, is only useful for reconstruction and not novel view synthesis, without any form of pose supervision. However, with the GAN approach, there are several limitations and next steps to consider that could be taken to improve our results.

Firstly, as can be seen in figure 11, our GAN inversion still suffers from many artifacts for reconstruction. We believe this is due to our naive approach to GAN inversion, which converges to a point where there is still significant reconstruction loss between an input image and G(z). For future work, we would use a more sophisticated encoder architecture for GAN inversion, such as the one that has been proposed for StyleGAN inversion [16].

As a next step, we would also train the GAN and GAN inverter auto-encoder style after training each component individually, to minimize end-to-toe reconstruction loss, and explore the effects of this step on the performance of our model.

Our GAN approach also suffers from relatively unstable training, and those chairs that are ultimately produced by the GAN still do not always resemble chairs 7. While we took several steps that are outlined in the analysis section to improve these results, there are still further improvements that could be made.

We lose a lot of information during rendered chair image transformation, both during render to sketch and sketch to inverted sketch. This results in our data being polluted by many bad samples, as can be seen in figure 12. As a next step, we could use non-photorealistic [7] rendering approaches to render our chairs sketch style, or use StyleGAN to perform a sketch style transfer on our rendered chair images and improve the quality of our training data.

We could also improve our GAN architecture itself, by replacing the DCGAN with a SketchGAN which also acts to fill in missing sketch information using multiple generators [10]. Alternatively, we could process our output frames using SketchGAN for sketch completion. This could reduce the effects of information loss during GAN training, and also improve the potential for reconstructing unfamiliar input data. Furthermore, SketchGAN is also used for classification, and could be a way for us to generalize our method to handle more than one object class.

Finally, our approach could benefit from disentanglement learning [21] to infer which portions of the latent code actually correspond to rotation. Currently, we rely on luck to obtain meaningful rotations from simple latent space interpolation. However, as we are training on synthetic data, we could supervise our training with pose information to infer which portions of the latent code representations are the most useful for rotation.

7. Conclusion

We present the first steps towards novel view synthesis for sketch images. We investigate the use of a GAN and a VAE for solving this problem, and demonstrate that a GAN and GAN inversion is more suitable for this task than a VAE. Our approach is able to infer meaningful rotations through latent space interpolation, although it suffers from information loss at several stages in the process. We conclude that the manipulation of the latent space of sketches is a promising direction for novel view synthesis and animation on line drawings.

References

[1] facebook. https://ai.facebook.com/blog/using-ai-to-bring-childrens-drawings-to-life. Accessed: 2021-12-16.

[2] VAE. https://colab.research.google.com/github/smartgeometry-ucl/dl4g/blob/master/variational_autoencoder.ipynb. Accessed: 2021-12-16.

[3] Jie Chen, Gang Liu, and Xin Chen. Animegan: A novel lightweight gan for photo animation. In *International Symposium on Intelligence Computation and Applications*, pages 242–256. Springer, 2019.

[4] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 279–288, 1993.

[5] Hojin Choi and Seungkyu Lee. Novel view synthesis from two cartoon face drawings. In *ACM SIGGRAPH 2017 Posters*, pages 1–2. 2017.

[6] Antonia Creswell and Anil Anthony Bharath. Inverting the generator of a generative adversarial network. *IEEE transactions on neural networks and learning systems*, 30(7):1967–1974, 2018.

[7] Bruce Gooch and Amy Gooch. *Non-photorealistic rendering*. CRC Press, 2001.

[8] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[9] Tejas D Kulkarni, Will Whitney, Pushmeet Kohli, and Joshua B Tenenbaum. Deep convolutional inverse graphics network. *arXiv preprint arXiv:1503.03167*, 2015.

[10] Fang Liu, Xiaoming Deng, Yu-Kun Lai, Yong-Jin Liu, Cuixia Ma, and Hongan Wang. Sketchgan: Joint sketch completion and recognition with generative adversarial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5830–5839, 2019.

[11] Lingjie Liu, Duygu Ceylan, Cheng Lin, Wenping Wang, and Niloy J Mitra. Image-based reconstruction of wire art. *ACM Transactions on Graphics (TOG)*, 36(4):1–11, 2017.

[12] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7588–7597, 2019.

[13] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021.

[14] Paul Rademacher. View-dependent geometry. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 439–446, 1999.

[15] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[16] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2287–2296, 2021.

[17] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database: learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (TOG)*, 35(4):1–12, 2016.

[18] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *arXiv preprint arXiv:2007.02442*, 2020.

[19] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019.

[20] Paul Wells. *Understanding animation*. Routledge, 2013.

[21] Jimei Yang, Scott E Reed, Ming-Hsuan Yang, and Honglak Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In *Advances in neural information processing systems*, pages 1099–1107, 2015.