ART AND MACHINE LEARNING
CMU 2019 SPRING
FINAL PROJECT

Interactive CPPNs

Zach Saffran, Angel Hernandez, Christopher George, Josh Moavenzadeh, Pancho Cabrera

1. Concept

In this class we sought to bridge two dissonant fields by creating art, a task often perceived as exclusively human, with machine learning algorithms. Many different approaches throughout the semester produced disturbing or disfigured results unappealing to the eye. Not only did results fail to be visually appealing, but they failed to connect with the viewer as a human due to their obscure and procedural nature.

Our project connects the user more intimately with machine learning art by allowing them to have agency over the output and by immersing them in a holistic audiovisual experience using CPPN-generated visuals and autoencoder-generated sounds.

2. Background

2.1 Compositional Pattern Producing Networks

Some of the original research on CPPNS [1] was in the realm of augmenting the topology of neural networks using the genetic algorithm. Neural networks would begin with a simple architecture and rather than using back propagation to update the weights, the network would use the genetic algorithm to not only update the weights but also the topology of the network, i.e., add more layers, hidden units and change activation functions. This has lead to many interactive web applications [2] where users can evolve patterns that start out from simple networks into more complex networks and visuals.

More recent work [3,4,5,6,8] have been using CPPNs to generate images. in its simplest form, a CPNN takes in a (x,y) coordinate and generates a (r,g,b) value at that location using a neural network with an arbitrary architecture. CPPNs can then be used to generate new images with N x N pixel locations and the weights in the network can be learned or simply random. At its core the job of the CPPN is to take in some representation of information and act as a generator to create the output, without actually producing a higher dimensional output for any single input, but a culmination of said inputs should generate a coherent set of outputs.

2.2 Neural Networks and Music

Traditional research has used RNNs and LSTMs [9,10] to generate music where these networks learn sequential aspects of training music to render new music. While these kinds of models have exhibited success, [11] proposes to use the decoder of of an autoencoder to render new music. Our project uses this is discussed in the methods section.

3. Exploration and Preliminary Findings

Since CPPNs were a new architecture to us and the visual component was a key aspect of our project, we played with different frameworks of CPPNs to find the best application.

3.1 CPPNs as a Prior in CNN Latent Visualization

Research is continuing to grow [3,5] in attempting to understand how hidden units in convolutional neural networks learn feature maps of images to obtain state of the art classification results and CPPNs have found a place in this research. Similar to DeepDream, [3] attempts to understand a hidden unit by maximizing its activation value by changing the input image via back propagation. Although, rather than using some arbitrary image as an input, a CPPN is used to generate an input image and the weights in the CPNN are learned to maximize the activation value of a given neuron in a CNN (VGG-19 for example). **Figure 1** in the Appendix showcases some images generated when attempting to activate different neurons in VGG-19. While some images had interesting aspects, the variability from image to image was hard to control and images did not take on a lucid free form we were going for.

3.2 Randomized CPPNs

[12] Uses a CPPN with randomized weights to generate rgb values across all pixel locations in a N1 x N2 image. While the network is simple, images take on interesting forms and are easy to control for variability using a latent vector **z.** The next section goes into detail of the model and **Figure 2** in the Appendix showcases some of the images generated in the preliminary stages. Ultimately, our group went with this CPPN framework because of its free form consistency across images and pattern/color controllability.

4 Methods and Techniques

4.1 Generating Image Decks

We began with the CPPN architecture layed out in [12]. The network is a FFNN with an arbitrary number of layers, units and activation functions that takes in a (x,y,**z,**d) tuple and returns a (rgb) value where (x,y) is the coordinate in the image, **z** is a controlling vector and d is the euclidean distance from the (0,0) coordinate in the image. **Figure 3** highlights how an image is generated using the CPPN architecture below.

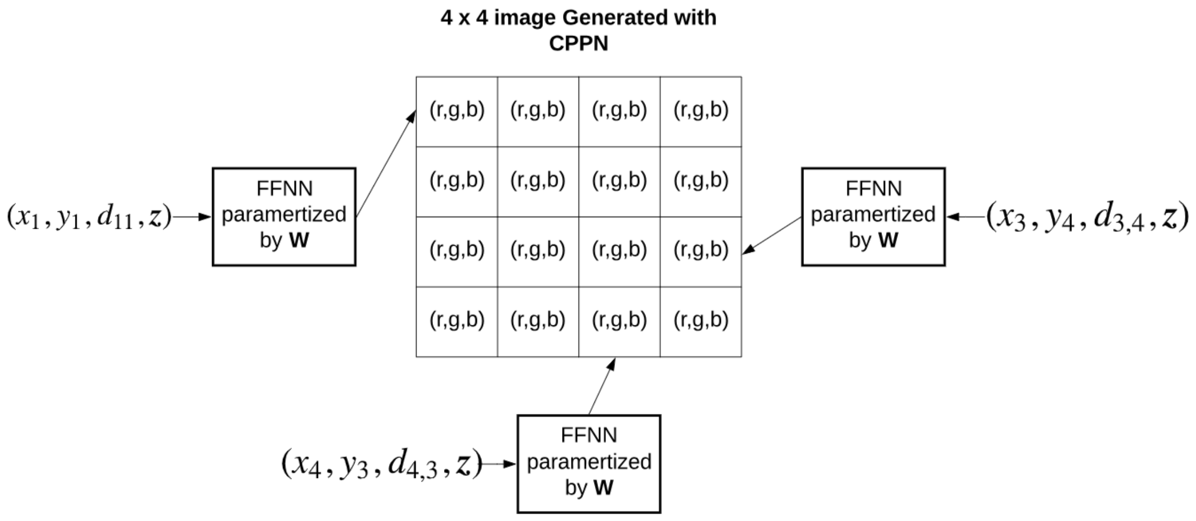


Figure 3: CPPN architecture used to generate a 4 x 4 image

Since the network is governed by an arbitrary set of weights, **W,** the vector, **z,** defines how images change from iteration to iteration where if **z** never changes, then you will always get the same image. So, we used different architectures to generate different image decks where images in the same deck took on similar spatial/color patterns because they all came from the same deterministic network, just a different **z** vector. **Figure 4** showcases two images from the same image deck.

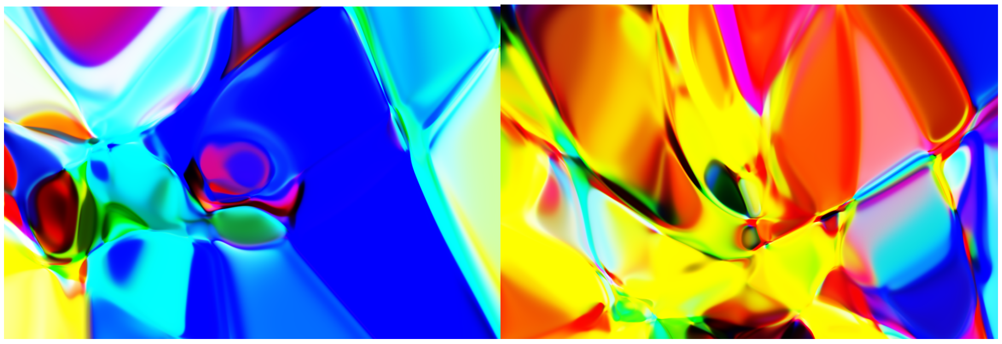


Figure 4: Two images created by the same network with different **z** vectors

4.2 Generating Music

We used an autoencoder neural network [12] and jazz/classical .midi files to learn and generate new music. During training, this network would take in a feature representation of a .midi file and forward propagate through two fully connected hidden layers to get an encoded representation of the music. Then, the encoded features are passed through two fully connected hidden layers which would decode the encoded feature vector to create a reconstructed song. This reconstructed song is trained to be identical to the input song but because the dimensionality of the encoder network is of a lower order, the model learns to create something similar while being original. After the model finished training, we would simply sample a latent vector and feed it into the learned decoder network to generate a new song. **Figure 5** highlights this training and music generating process.

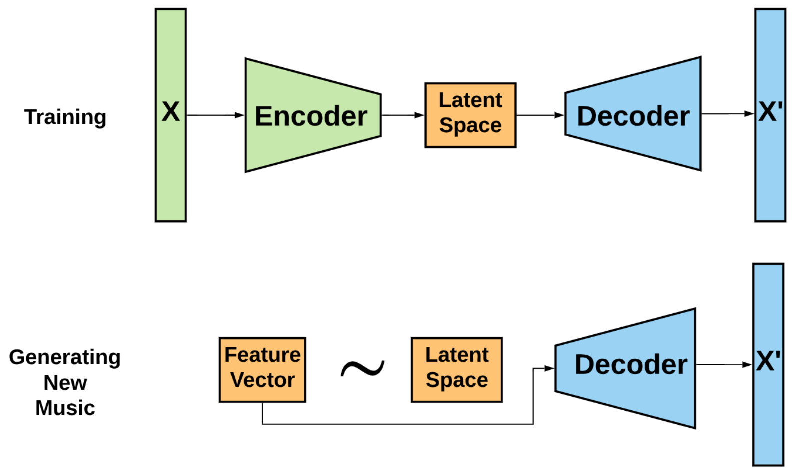


Figure 5: Processing for generating new music

4.3 Interactivity and Leap Motion

In order to bring an interactive component to the user we use a Leap Motion [13]. It is a device that interpolates hand movements into a coordinate present in a M-dimension space (where M is user defined).

4.3.1 Interactive Video (Left Hand)

We setup the leap motion where the left hand of the user is mapped to a 2-dimensional coordinate which controls the image that is rendered in real-time. This is achieved by taking two image decks and assigning each deck to a dimension. So, as a user moves their hand over the leap motion, they are being mapped to two images in different decks and we use min, max and different value functions to blend those images into a single image. This is highlighted in the **Figure 6.**

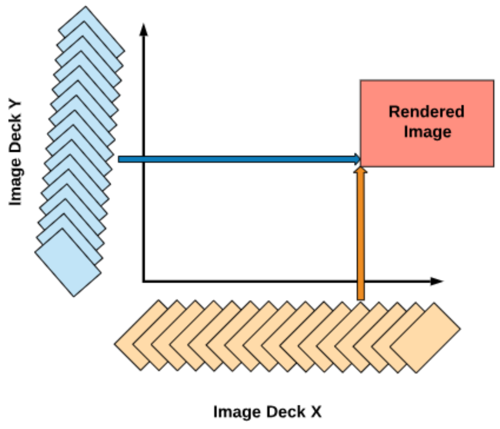


Figure 6: Mapping two images across different dimensions

Ultimately, this leads to a real-time video which cycles through different images that are mapped from the user's movement.

4.3.1 Interactive Music (Right Hand)

The right hand of the user has control over the music where each fingertip is mapped to a (x,y,z) coordinate where each dimension in the coordinate controls one index of the feature vector feed into the decoder network; this gives the user control over 15 features used to generate music. As the user moves their right hand they are changing the feature vector used to generate music which in turn changes the song in real-time.

5 Results

Presentation Video (High Quality):
https://cmu.box.com/s/cfttupsy5dtv6uskywn32ygy9r46imhi
Presentation Video (Low Quality):
https://www.youtube.com/watch?v=J29quJOoq8M&t=66s
Interactive Music + Randomized CPPN Video (High Quality):
https://cmu.box.com/s/ew2jc5k4vexw6kzjvq2wu9q6m2r6dmom
Interactive Music + Randomized CPPN Video (Low Quality):
https://www.youtube.com/watch?v=HPNtFcdIFKY&feature=youtu.be

Some HD Image Results are below the ***References*** section.
Additionally, some photos from the presentation are also below the ***References*** section.

Code

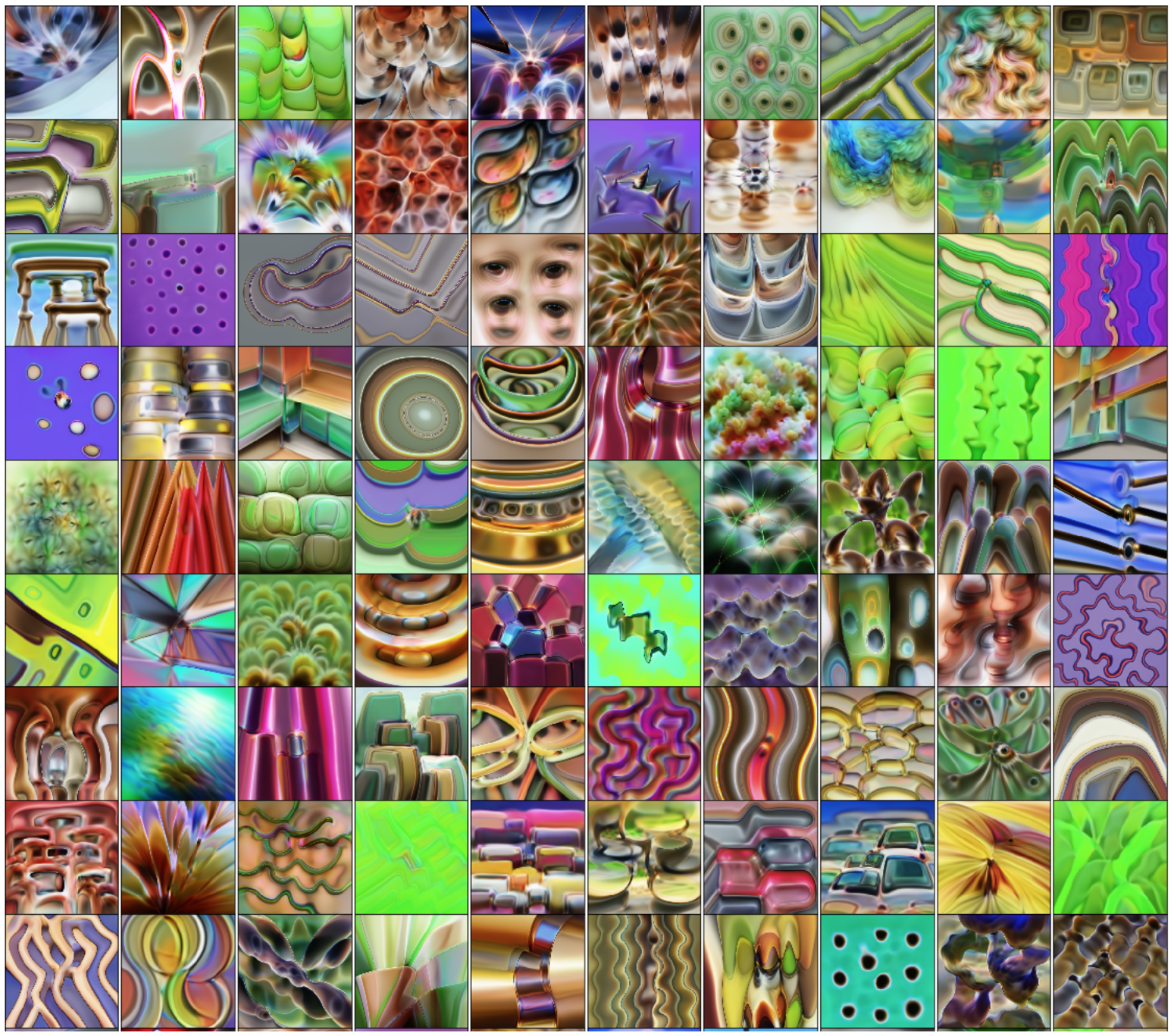
https://github.com/chrismgeorge/CPPN-Animation-Creator
https://github.com/chrismgeorge/Leap-Motion-CPPN-Interaction

5 Conclusion and Future Work

The combination of Neural Networks combined with human interaction to create art pieces is a field of work that we would like to continue to explore and be apart of. It is exciting to be in the AI and Art field that is exploding at this very moment. Although we have explored CPPNs as much as we could in the time we had for this project, we may find interest in continuing that exploration. Trying to find architectures, or other processes for making our images possibly less abstract and more structured in reality. Other work that we cited uses the CPPN concept in unique ways that involve training, and trying to fully understand how to implement those processes for our own uses may be interesting. Additionally the work that went into making the music for our project may, although interesting and fun, seemed to be lacking in overall tone, and complexity. Finding and researching ways to make better “fake music,” would be an interesting task to pair with these videos. Additionally, the overall idea, or making videos based on music or vice versa was something that we are also thinking about exploring. That concept was something we would have liked to explore for this final presentation but were unable to.

Appendix

Figure 1

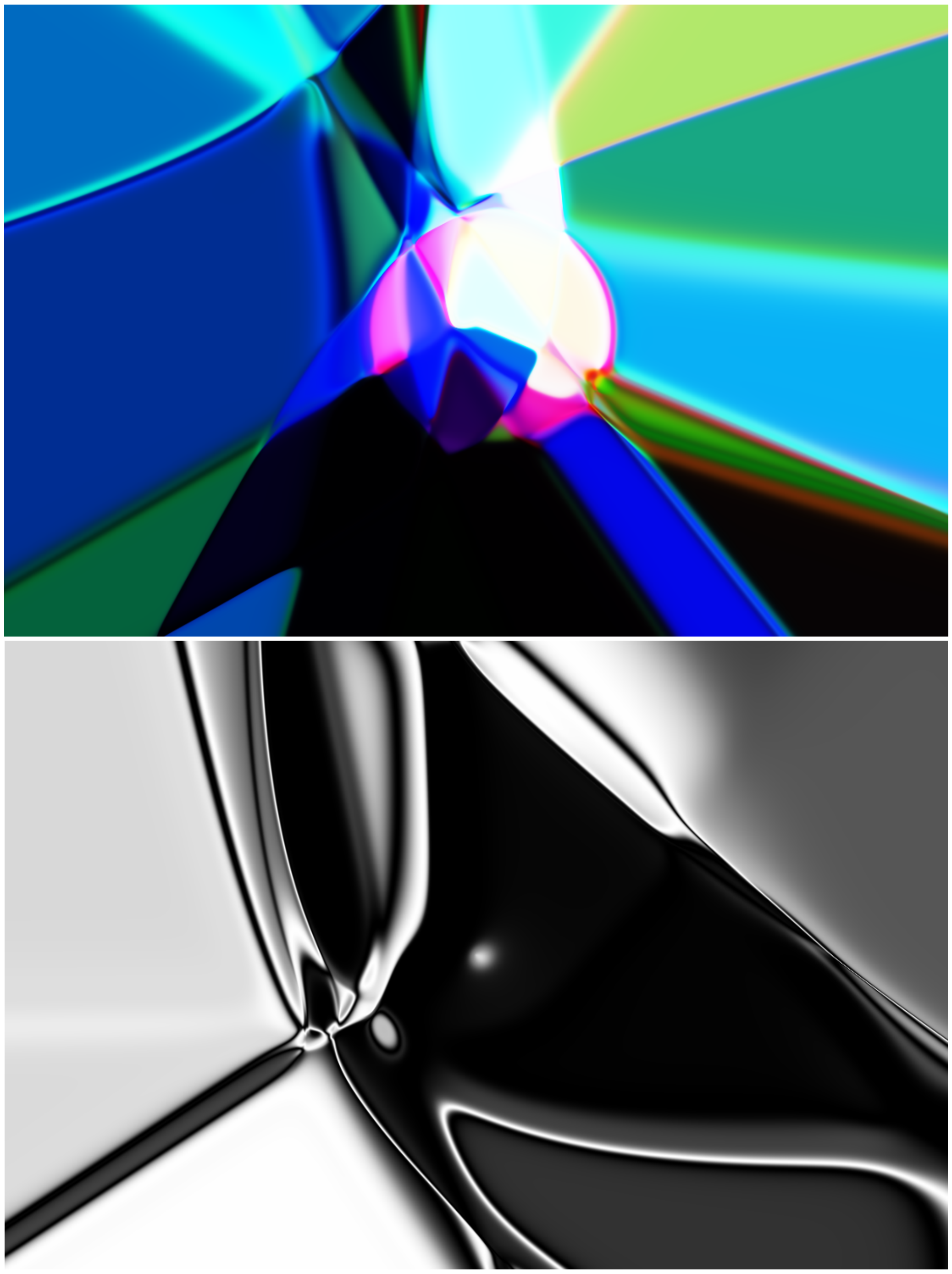


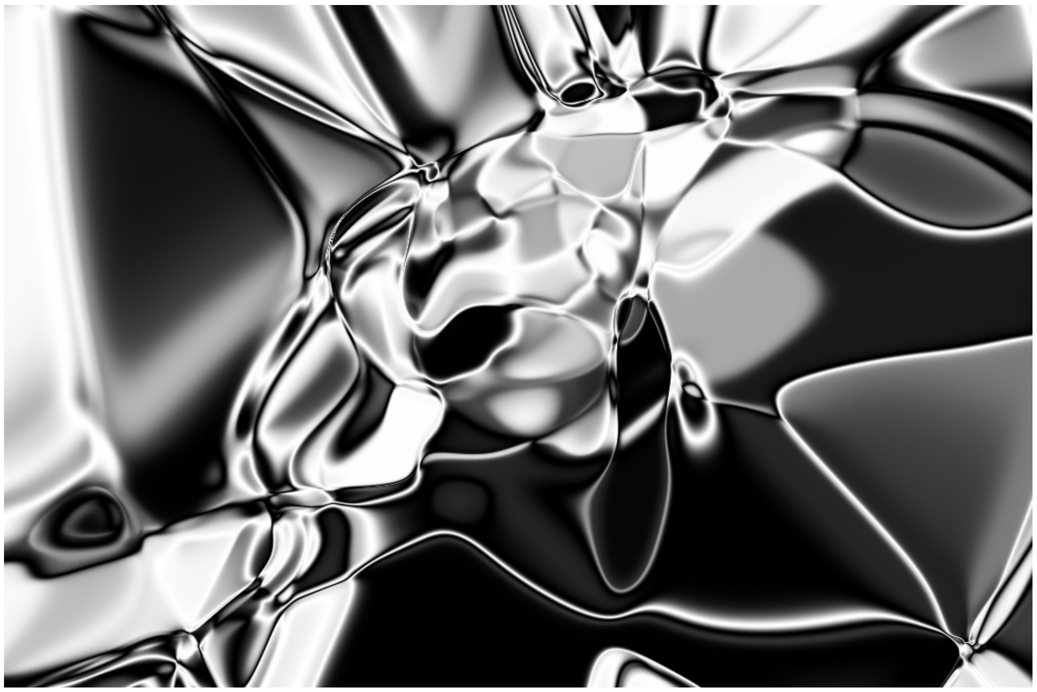
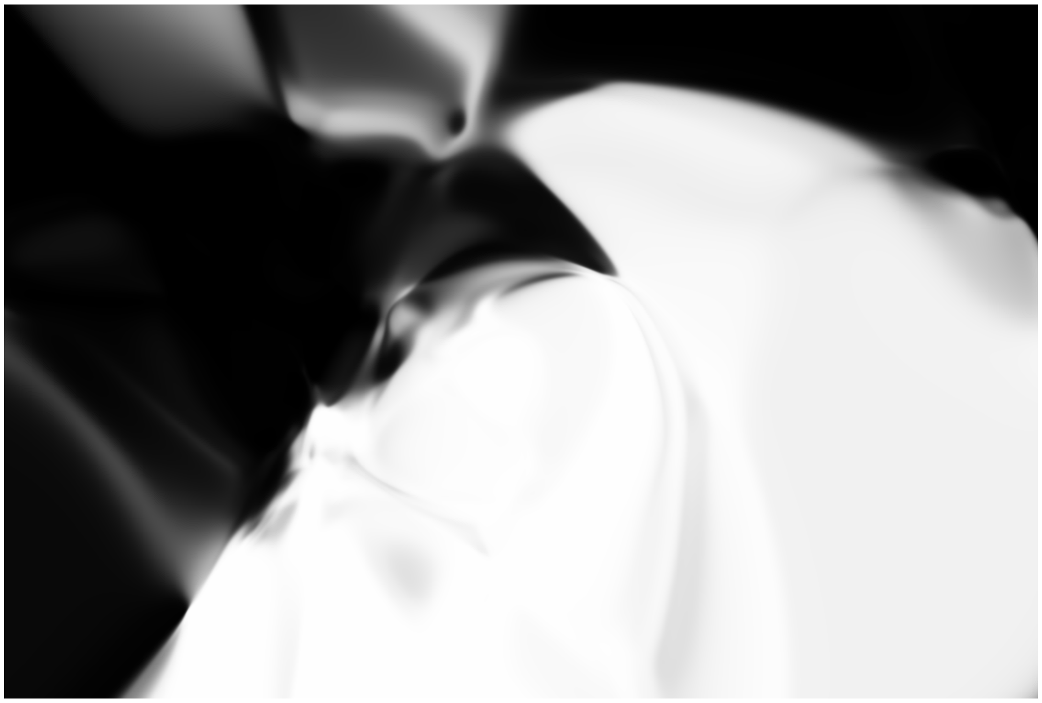
References

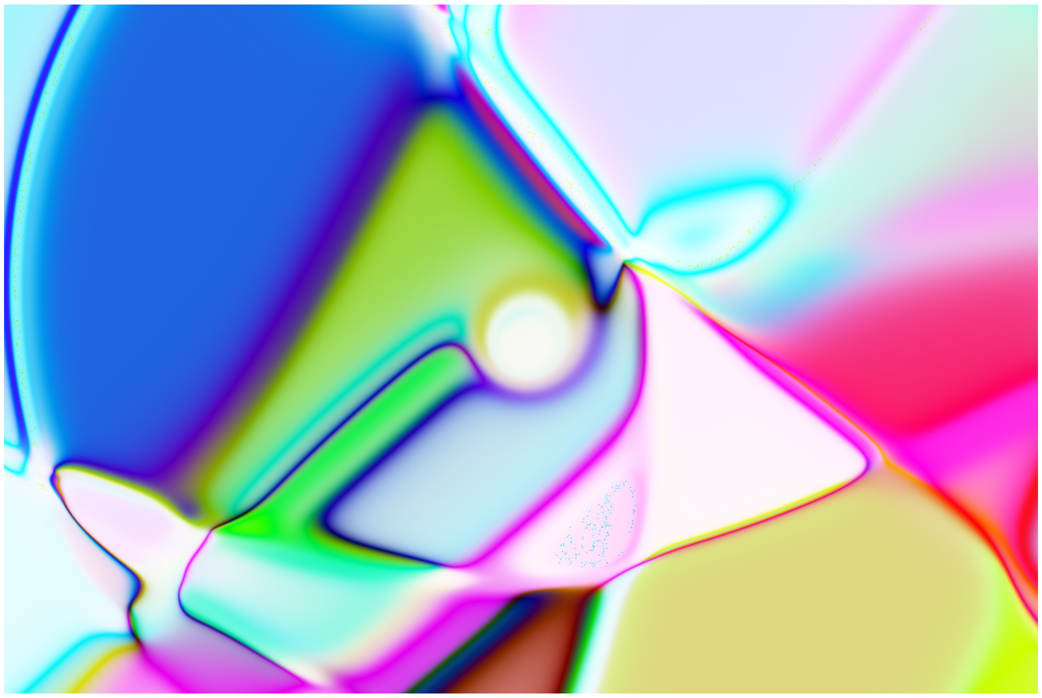
[1] https://eplex.cs.ucf.edu/papers/stanley_gpem07.pdf
[2] http://blog.otoro.net/2015/07/31/neurogram/
[3] https://distill.pub/2018/differentiable-parameterizations/#section-xy2rgb
[4] http://blog.otoro.net/2016/04/01/generating-large-images-from-latent-vectors/
[5] https://distill.pub/2017/feature-visualization/appendix/
[6] http://blog.otoro.net/2016/03/25/generating-abstract-patterns-with-tensorflow/
[7] https://github.com/tensorflow/lucid
[8] https://janhuenermann.com/blog/abstract-art-with-ml
[9] https://magenta.tensorflow.org/demos
[10] https://cs224d.stanford.edu/reports/allenh.pdf
[11] https://github.com/HackerPoet/Composer

[12]http://blog.otoro.net/2016/03/25/generating-abstract-patterns-with-tensorflow/?fbclid=IwAR0t5JShT_QCAa7Hxkkq2VPLhGSwBVaKD3AoqUh3Kz88gkX5JhwUk-M961s
[13] https://en.wikipedia.org/wiki/Leap_Motion

HD Image Results





Presentation Images (People viewing / interacting with the piece)