From Sketch to 3D

Team: Yufeng Zhang, Bihao Zhang, Haotian Zhang

Advisor: Prof. Hod Lipson

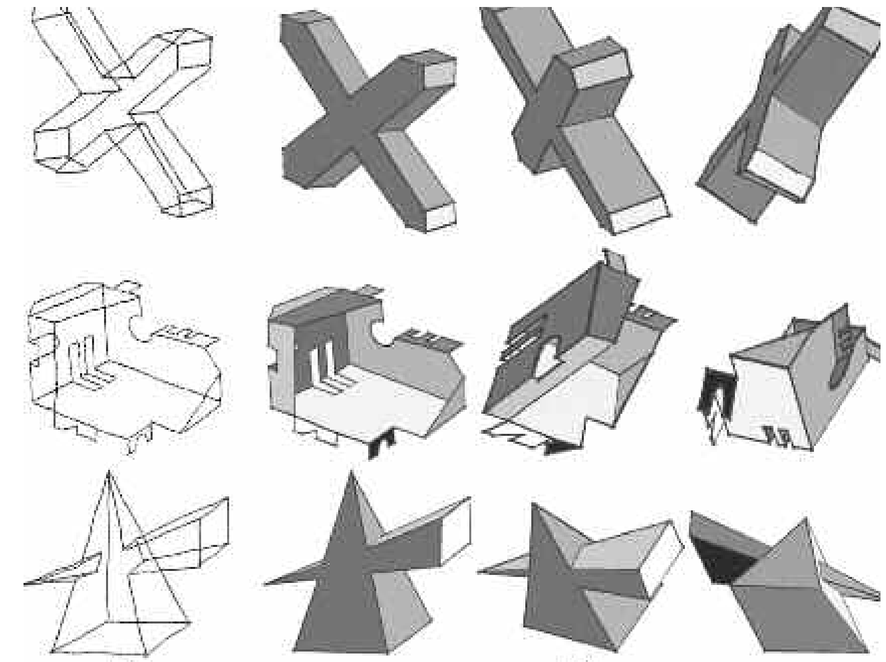
2017.12.11

Contents

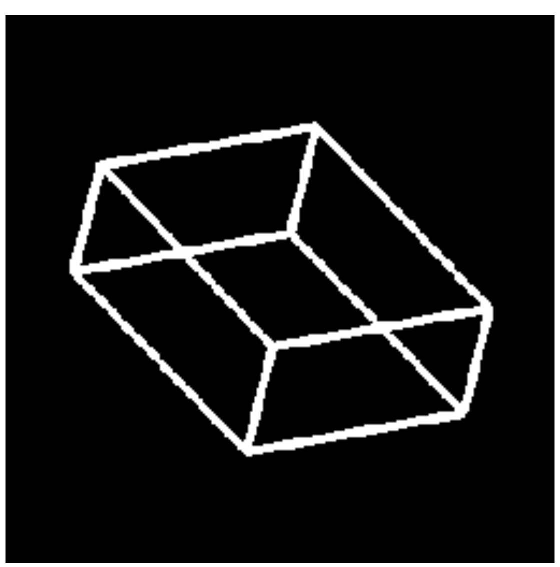
- Introduction to problem and models
- Training and testing
- Evaluation
- Dataset

What is the goal?

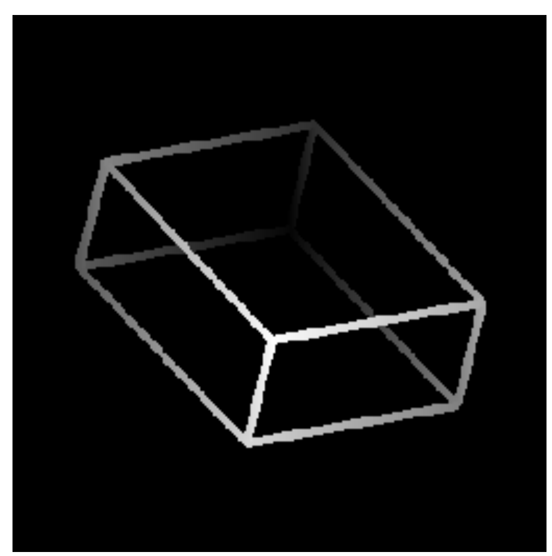
- Learning to generate 3D shapes from sketches



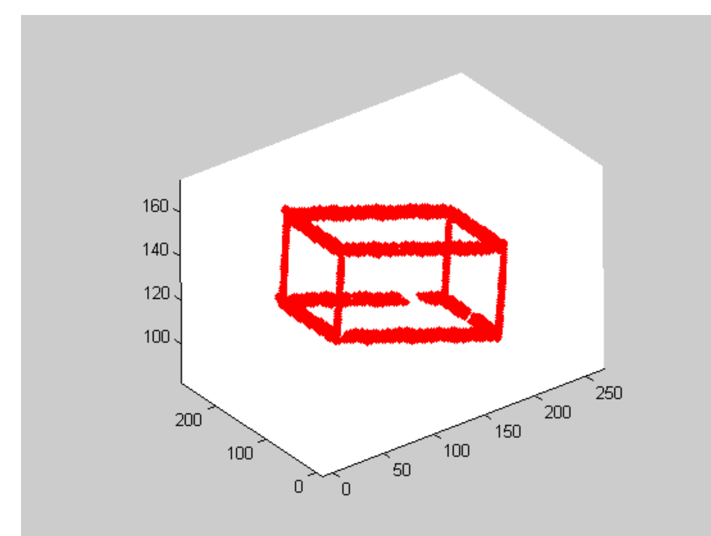
- Example: Cuboid



Sketch:
Black-white image



Depth image:
Grayscale image

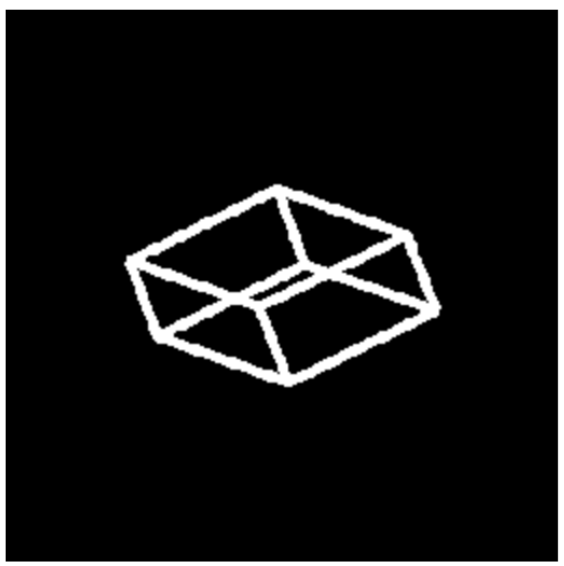


3D Shape:
Points in 3D space

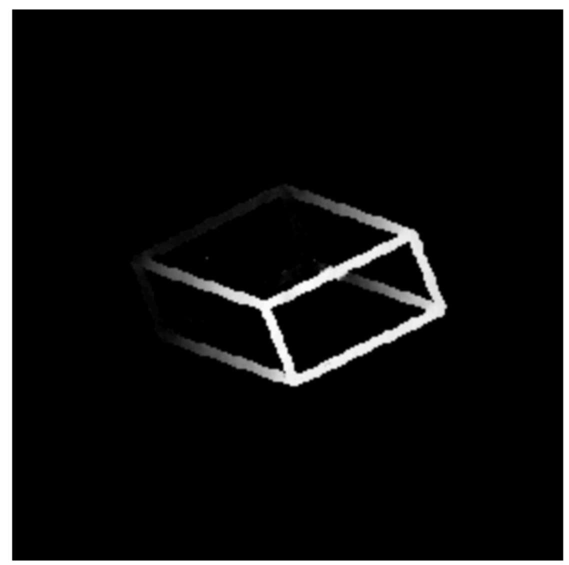
Deep learning

Graphics

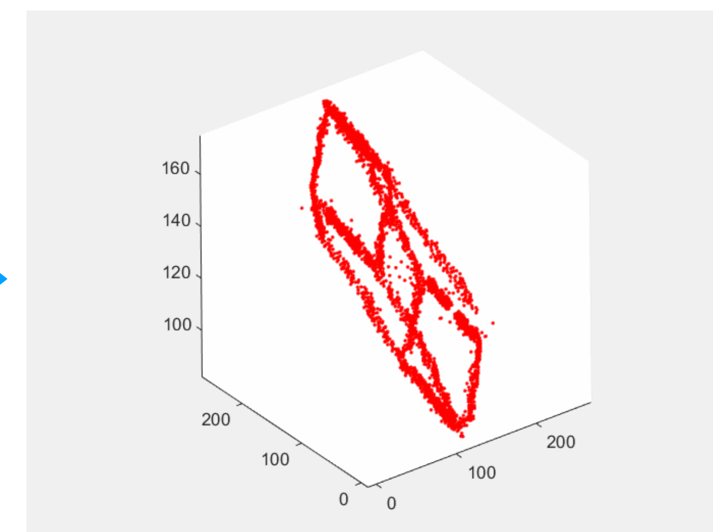
Current progress



Sketch:
Black-white image



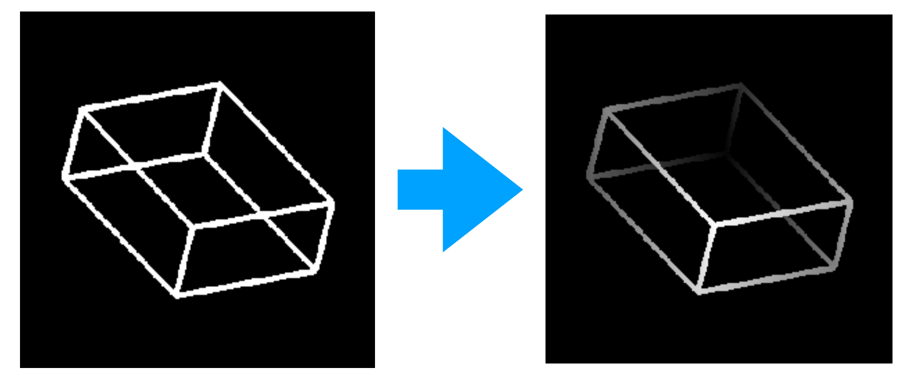
Depth image:
Grayscale image

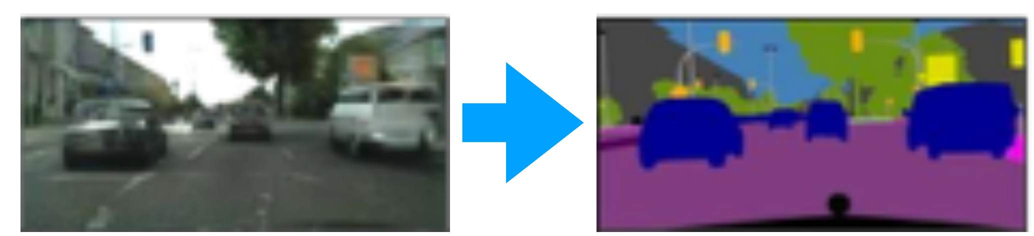


3D Shape:
Points in 3D space

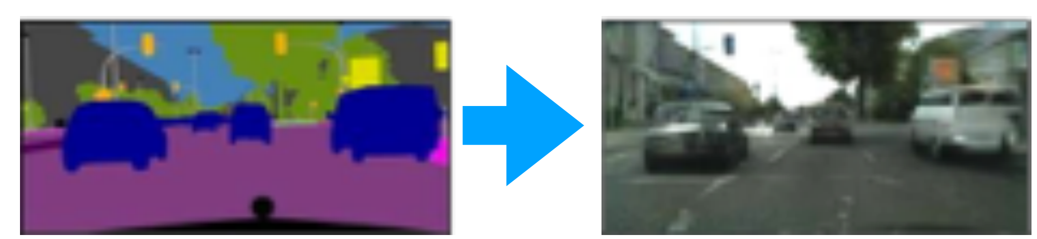
How to model the problem?

- Model as a segmentation problem
    - A classification problem, but the workflow is counter to the common workflow
    - The information of inputs is too limited (only black and white pixels)
    - The classes of the output are too many (256 classes)
- Model it as a generative problem
    - Conditional Generative Adversarial Networks (cGANs)





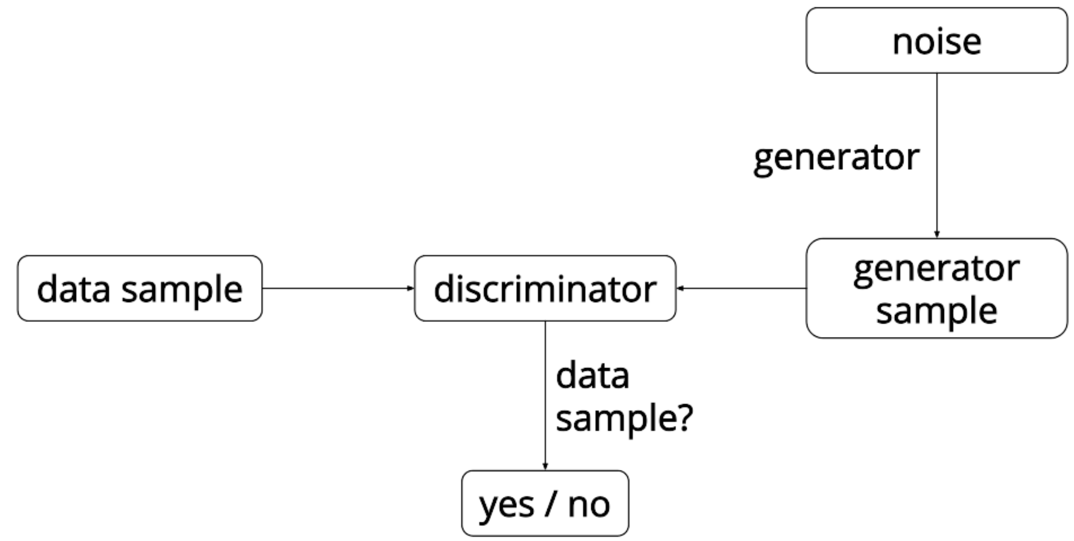
Workflow of traditional segmentation problem

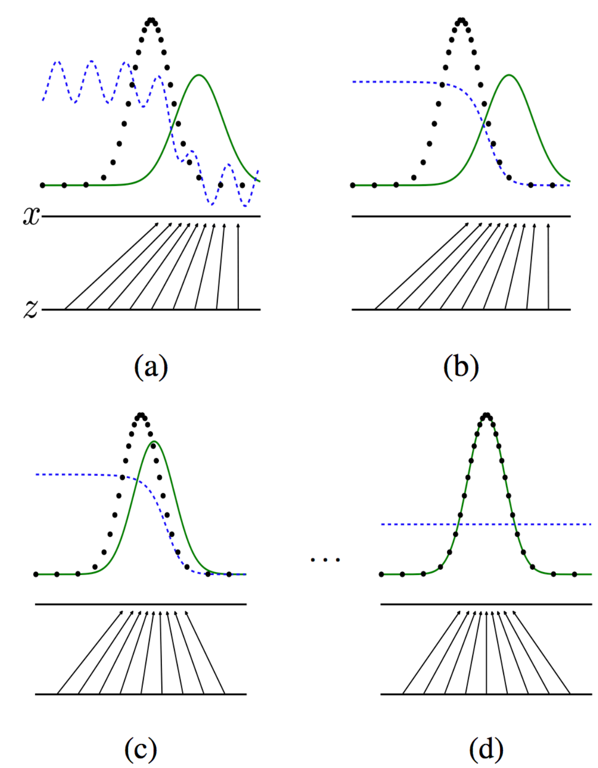


Analogy: Potential workflow of our problem
It is more similar to a generation task.

Generative Adversarial Networks（GAN）

- Objective: learn to generate data similar to given data
- Main idea: two competing neural network models
    - Generator: take noise as input and generate data
    - Discriminator: receive samples both from generator and training data, has to be able to distinguish between the two sources
- Training:
    - Two models are trained simultaneously
    - The hope is that the competition will drive the generated samples to be indistinguishable from real data





Blue, dashed line:
Discriminative distribution

Green, solid line:
Generative distribution

Black, dotted line:
Sampled data distribution

x: domain of generated data,

x = G(z)

z: domain of sampled data

"Generative Adversarial Networks." Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. ArXiv 2014.

Conditional GAN (cGANs)

- Difference: both generator and discriminator observe an input image
- Learn a mapping from observed image x and random noise vector z to y:
- y = f(x, z)

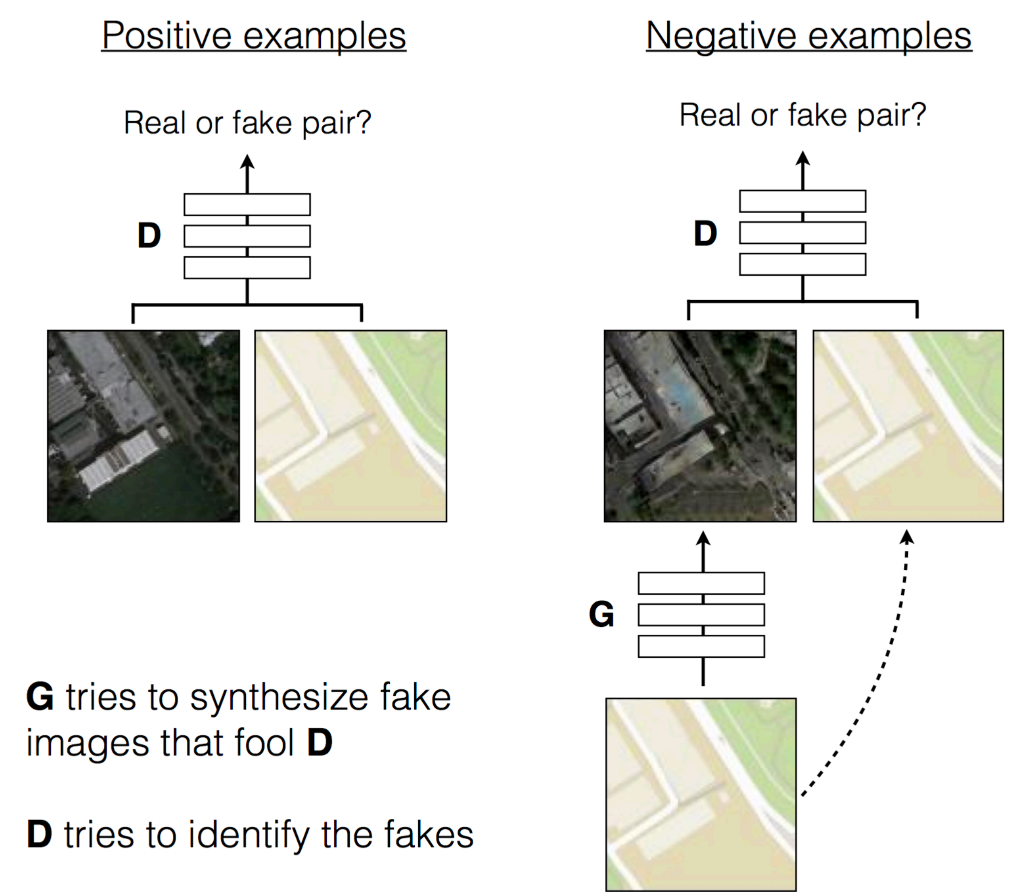


Image-to-Image Translation with Conditional Adversarial Networks, Isola, Phillip and Zhu, Jun-Yan and Zhou, Tinghui and Efros, Alexei A, arxiv, 2016.

Examples using cGANs

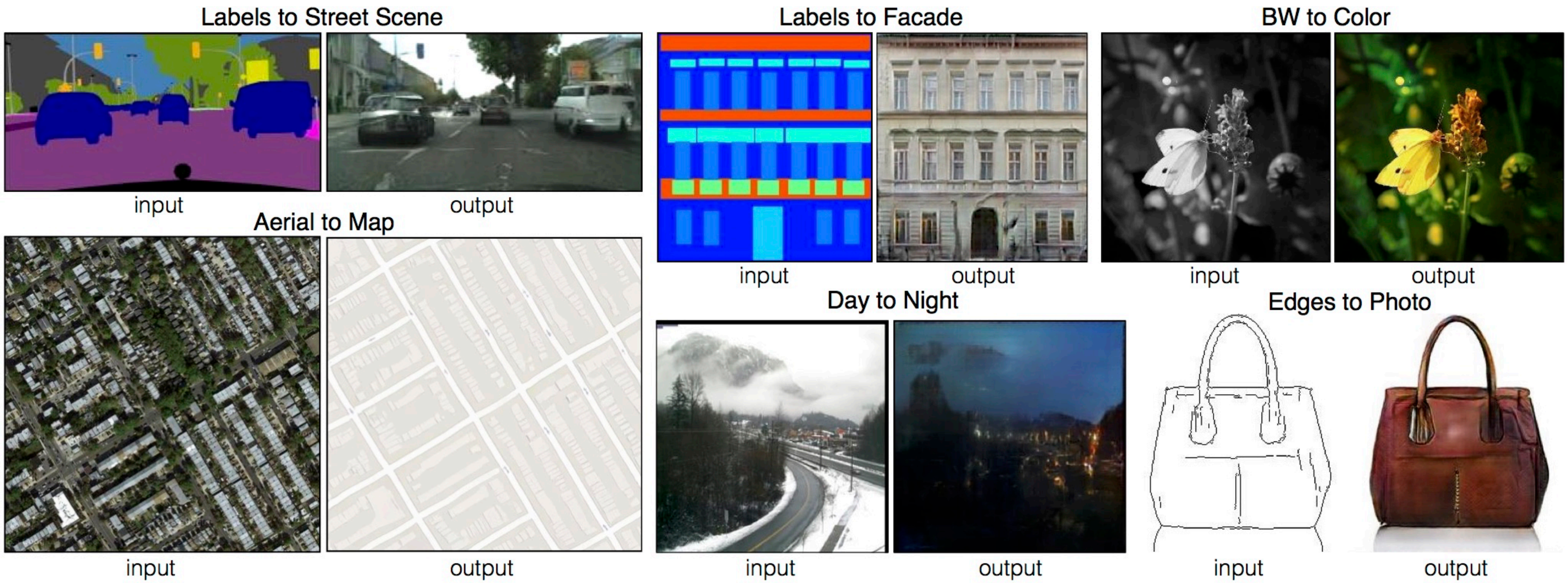
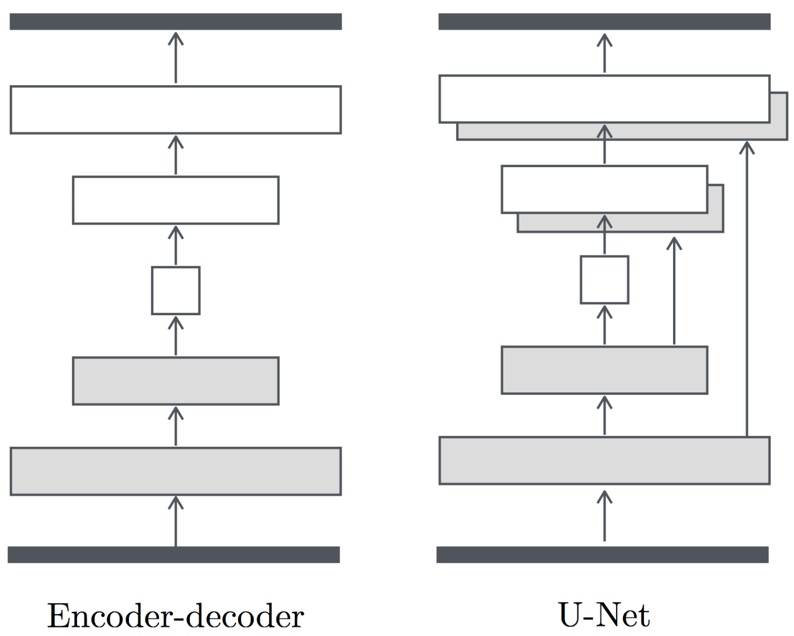
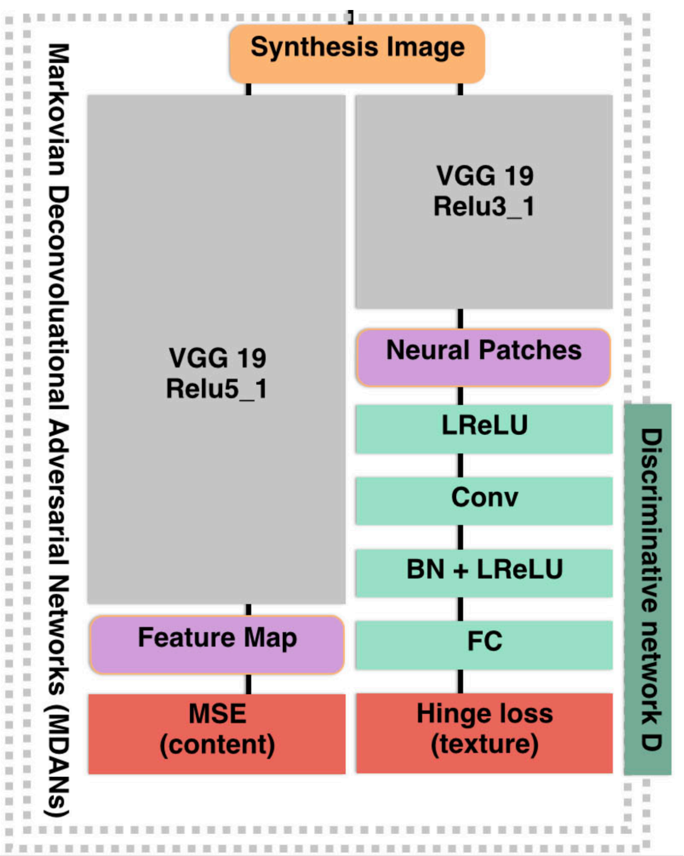


Image-to-Image Translation with Conditional Adversarial Networks, Isola, Phillip and Zhu, Jun-Yan and Zhou, Tinghui and Efros, Alexei A, arxiv, 2016.

Architecture of cGANs

- Generator
  - Encoder-decoder
  - U-Net: connections between mirrored layers in the encoder and decoder stacks
- Discriminator
  - Markovian discriminator (PatchGAN)
  - Convolutional discriminator on each patches of image

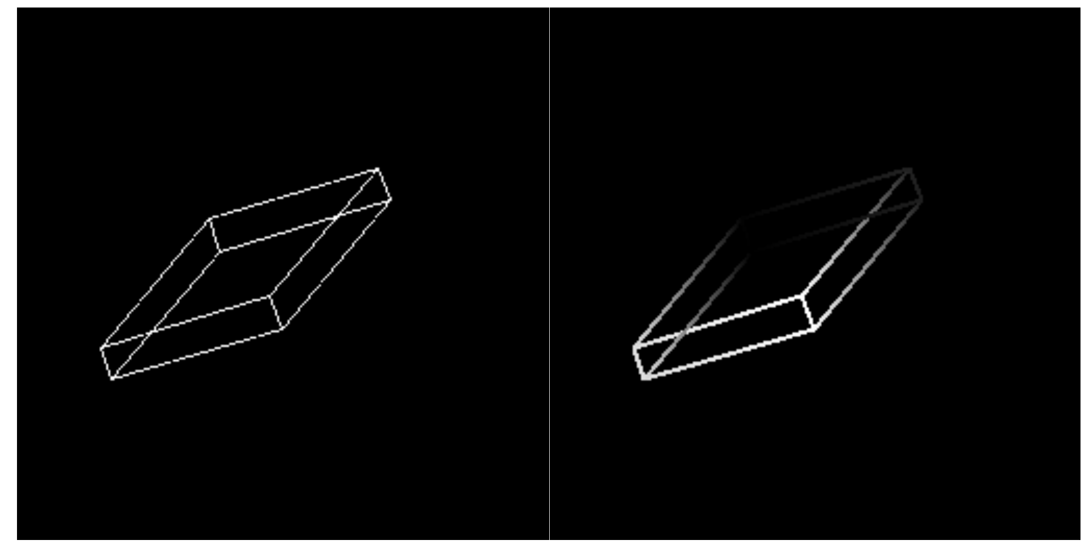


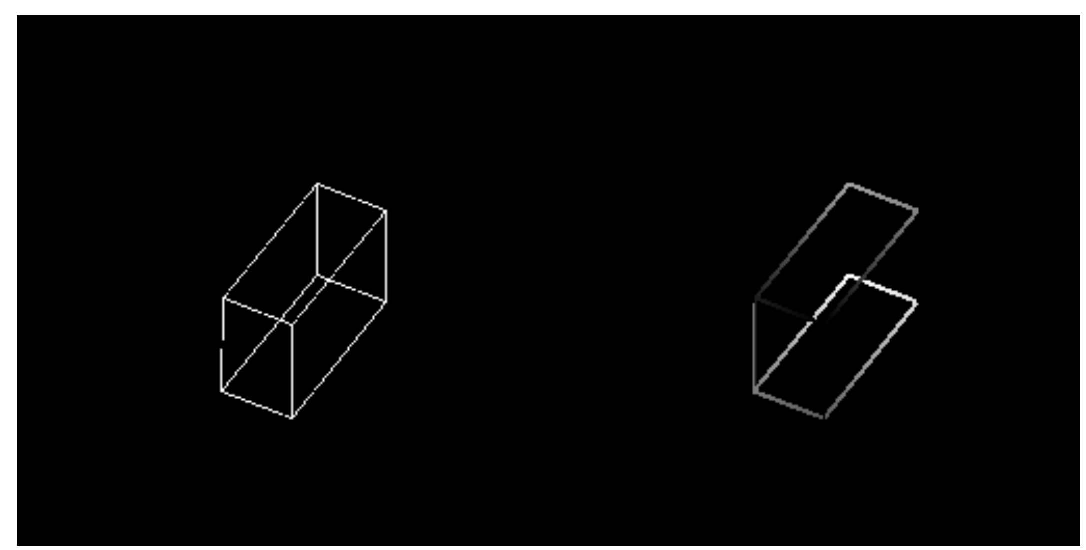


Li C., Wand M. (2016) Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016.

Training on conditional GAN

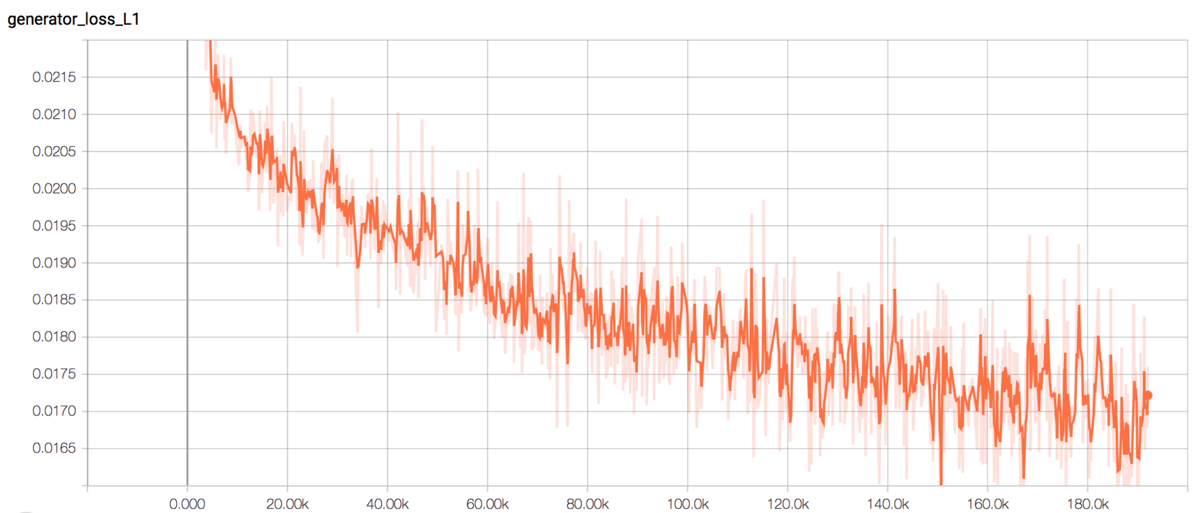
- Training data
  - Combined images of sketch and target image with depth information
  - Size: 256*512
  - Quantity: 1600
- Hyper parameters
  - Epoch: 120
  - Learning rate: 0.002
  - Batch size: 1
- Time consumed: 12 hours



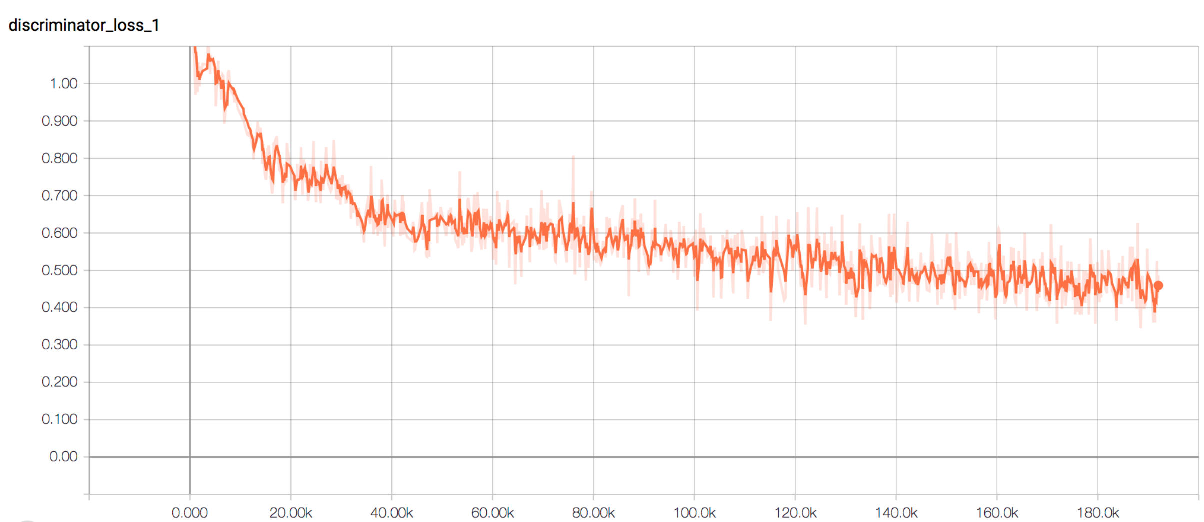


Training on conditional GAN

- Generator L1 Loss:
  - $\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y \sim p_{data}(x,y), z \sim p_z(z)}[\|y - G(x,z)\|_1].$
  - Reduced to 0.017

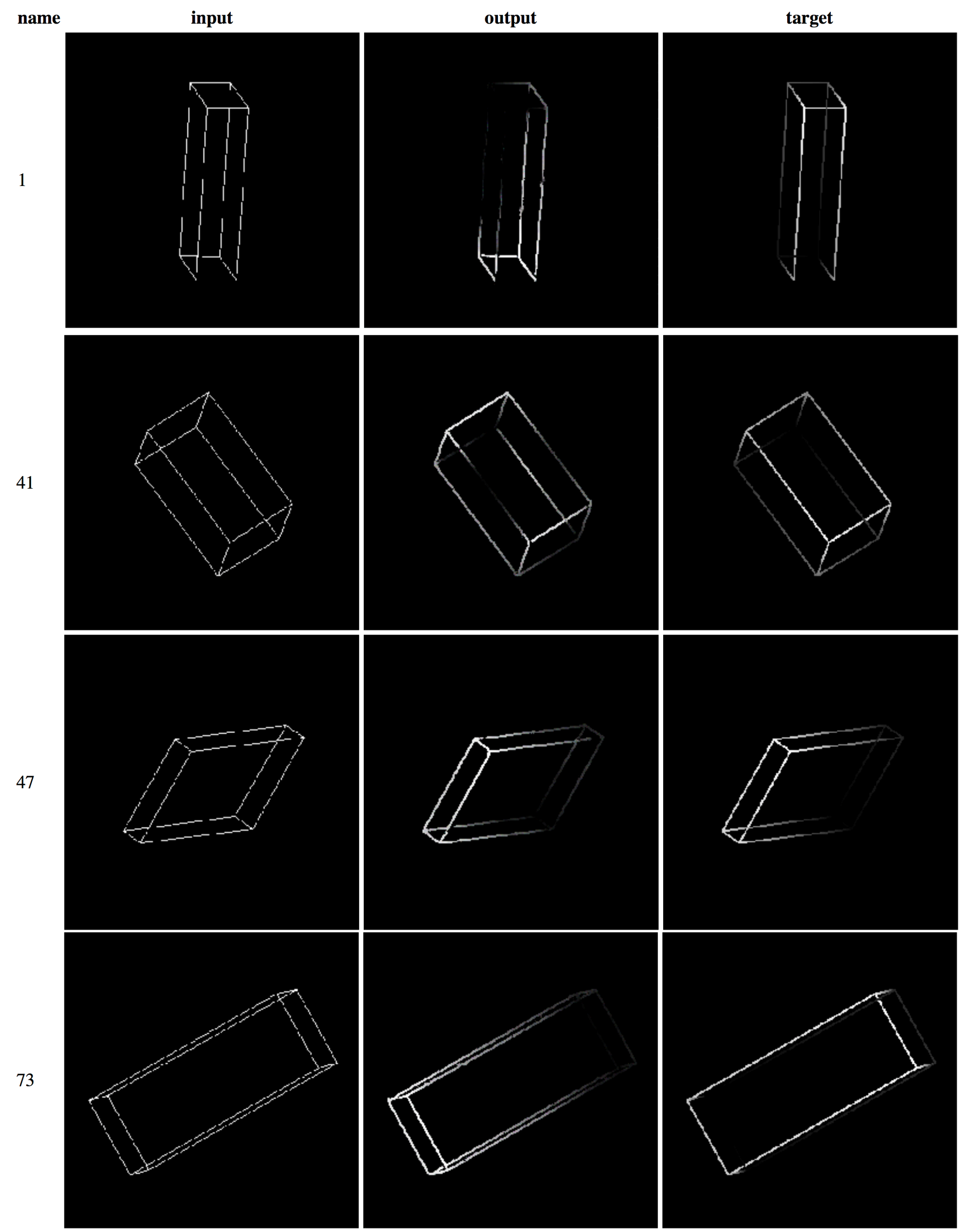


- Discriminator Loss:
  - Reduced to 0.459



Testing on conditional GAN

- Test data (left column)
  - Images of sketch
  - Size: 256*256
  - Quantity: 400
- Output (middle column)
  - Generated images with depth information
- Target (right column)
  - Ground truth images with depth for each sketch



Questions observed

- The depth of the points closed to the vertex is occasionally discontinuous, as shown in No.1

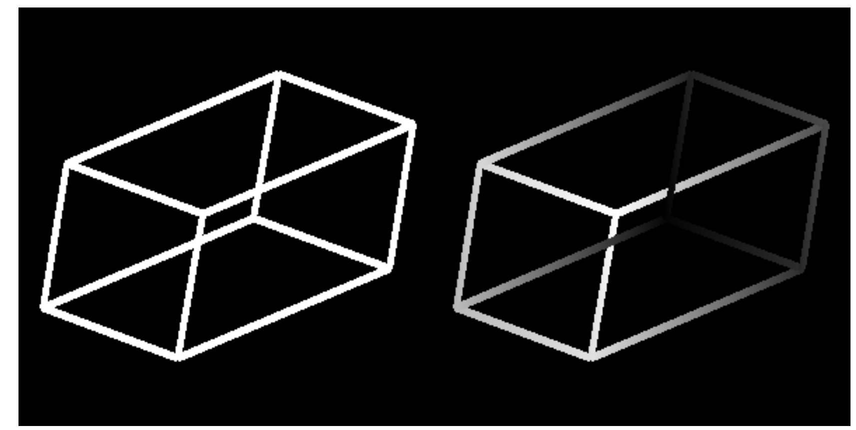
- The depth of the points on the same edge is not monotonically changed, as shown in No.41

- Edges of shape in each input image is so thin that they may become discontinuous after the image is resized, as shown in No.47

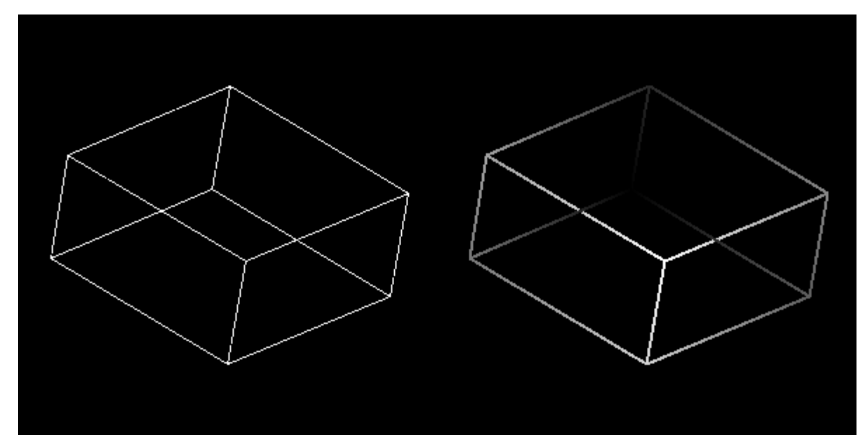
- Output image may randomly become one of the two possible depth images of each sketch (which indicates we need to update our dataset), as shown in No.73

Update on dataset

- Bolded the edges of shape which is more suitable for resizing
- Enlarged the size to 5000 images
- Fixed several bugs



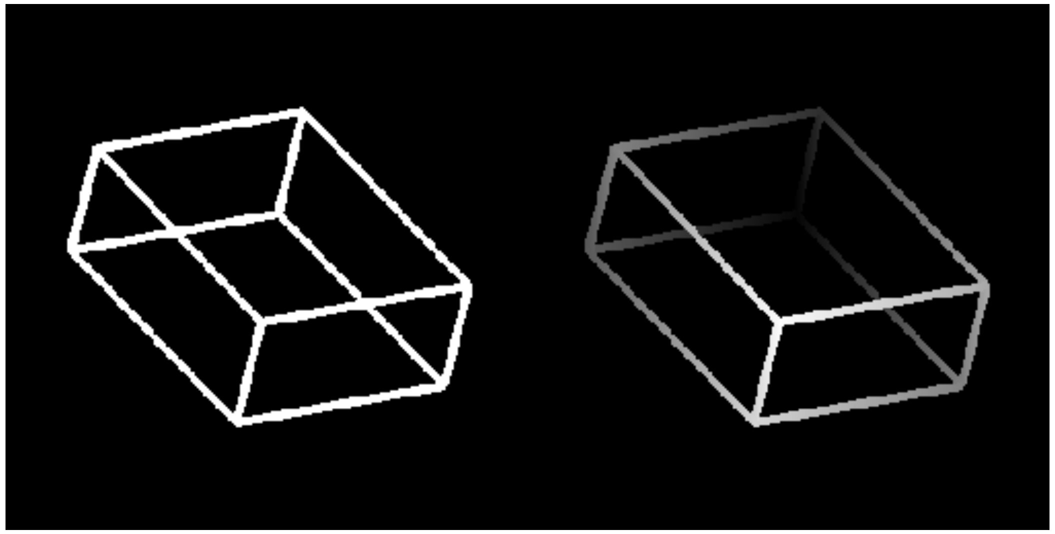
Sample of updated dataset

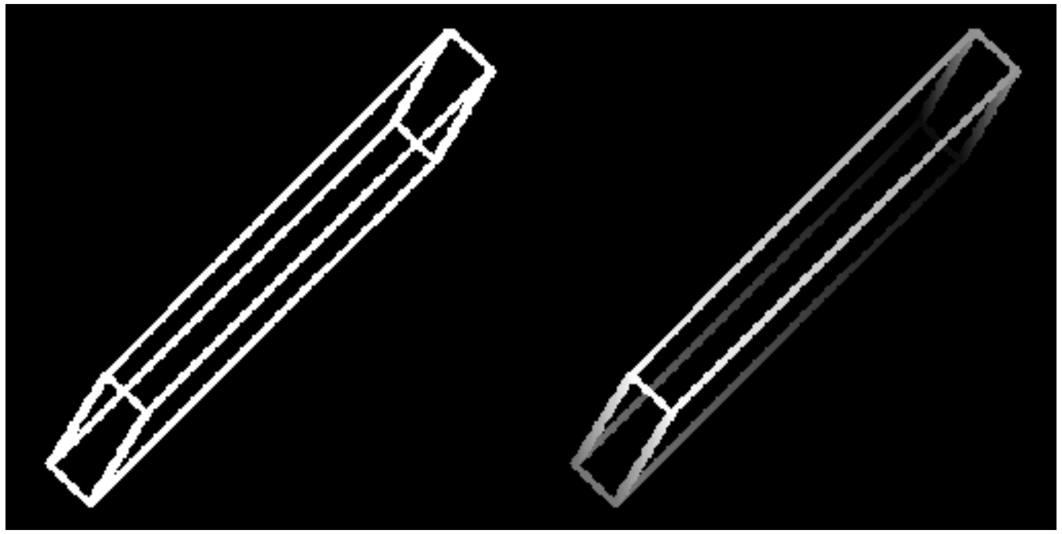


Sample of original dataset

Training on conditional GAN

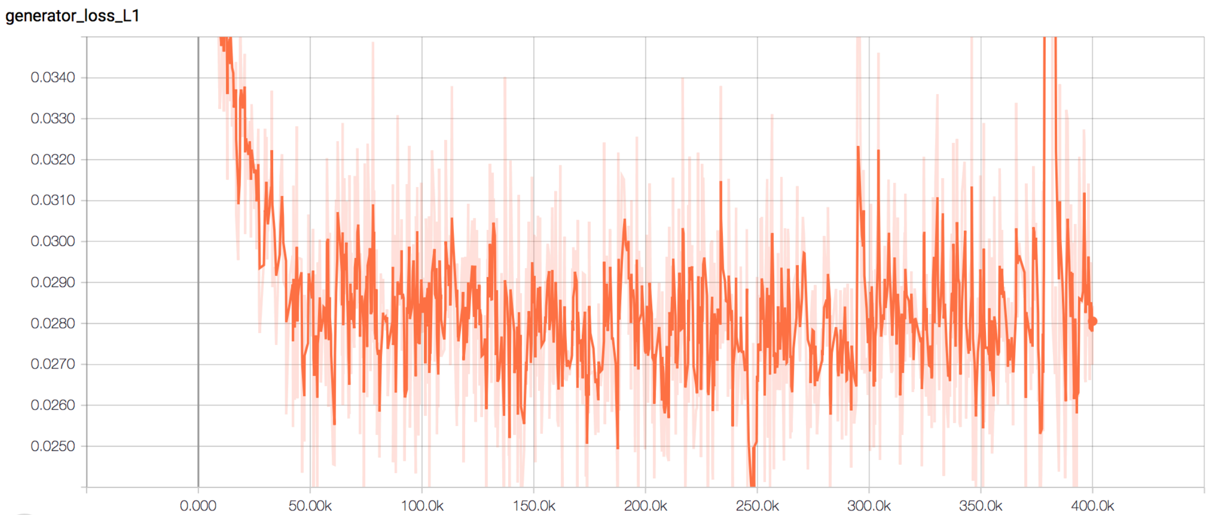
- Training data
  - Combined images of sketch and target image with depth information
  - Size: 256*512
  - Quantity: 4000
- Hyper parameters
  - Epoch: 100
  - Learning rate: 0.002
  - Batch size: 1
- Time consumed: 30 hours



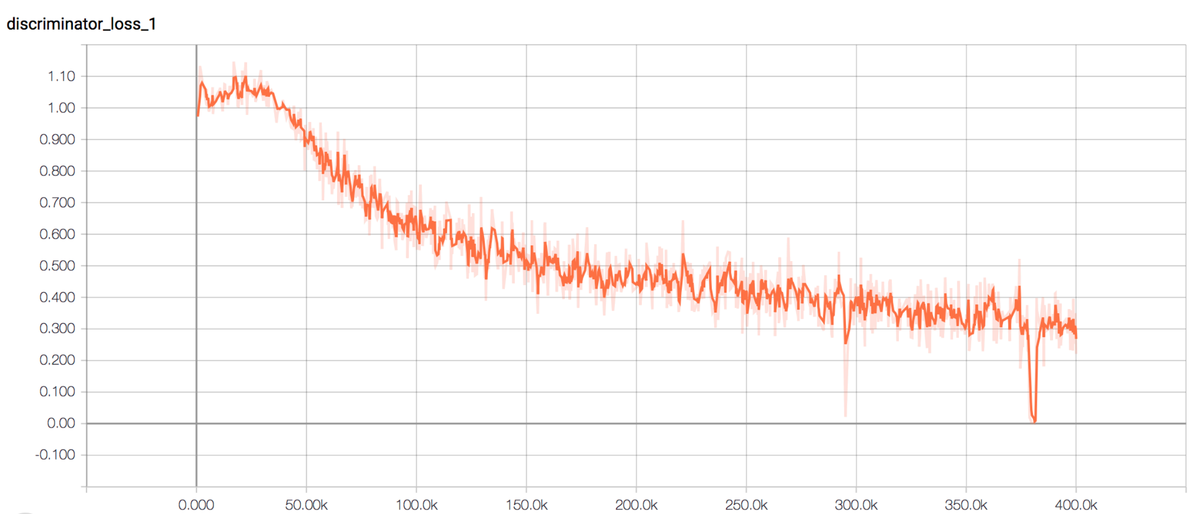


Training on conditional GAN

- Generator L1 Loss:
  - $\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y\sim p_{data}(x,y),z\sim p_z(z)}[\|y - G(x,z)\|_1].$
  - Reduced to 0.028

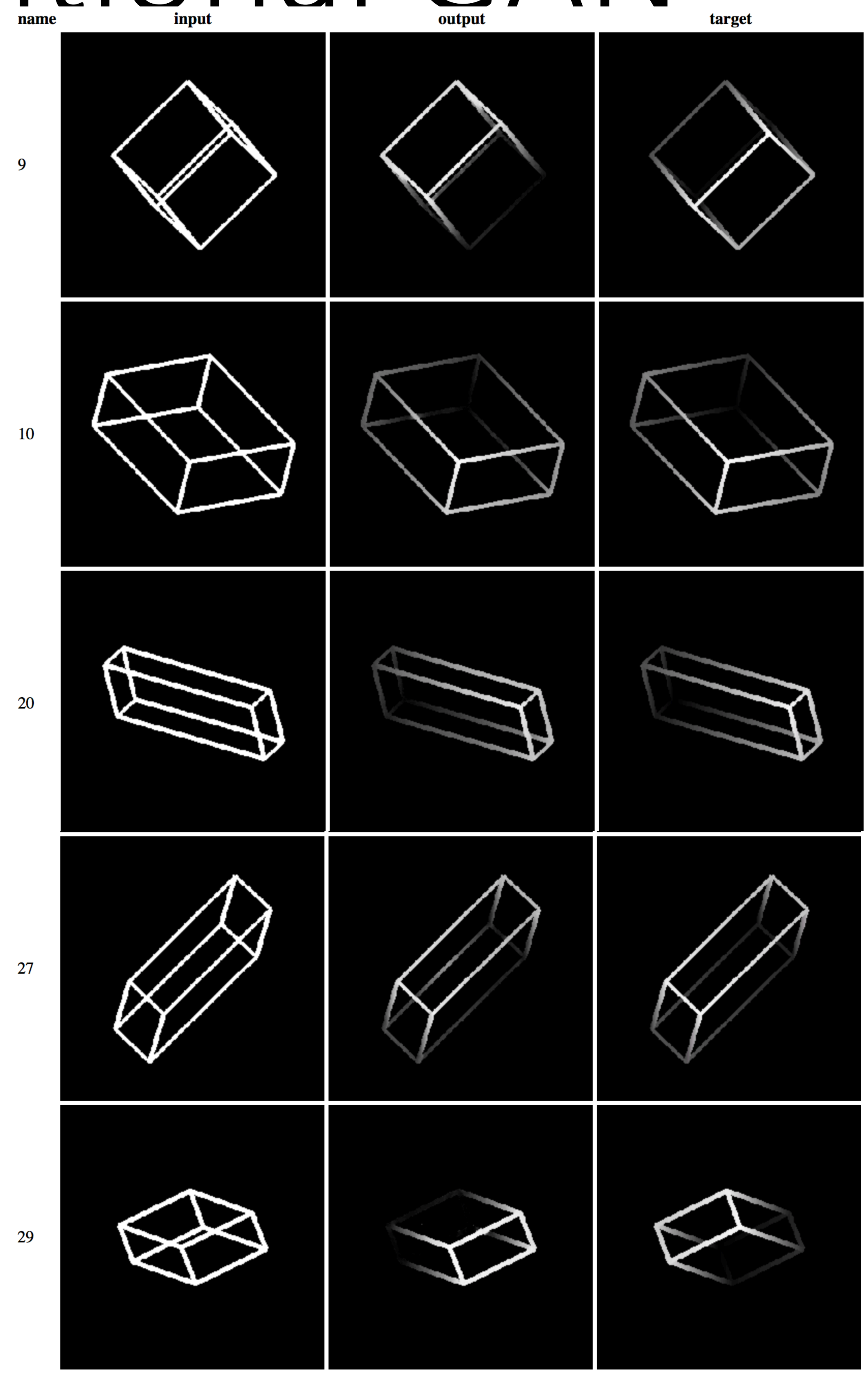


- Discriminator Loss:
  - Reduced to 0.283



Testing on conditional GAN

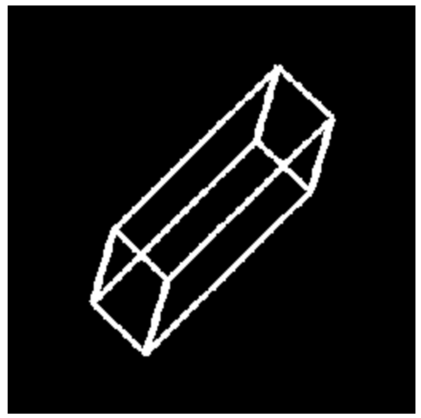
- Test data (left column)
  - Images of sketch
  - Size: 256*256
  - Quantity: 1000
- Output (middle column)
  - Generated images with depth information
- Target (right column)
  - Ground truth images with depth for each sketch

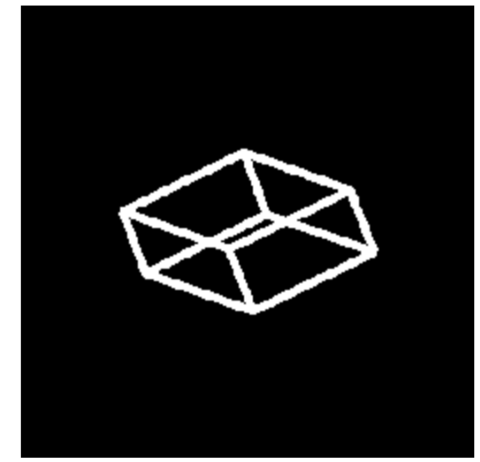


Results

Sketch:

27

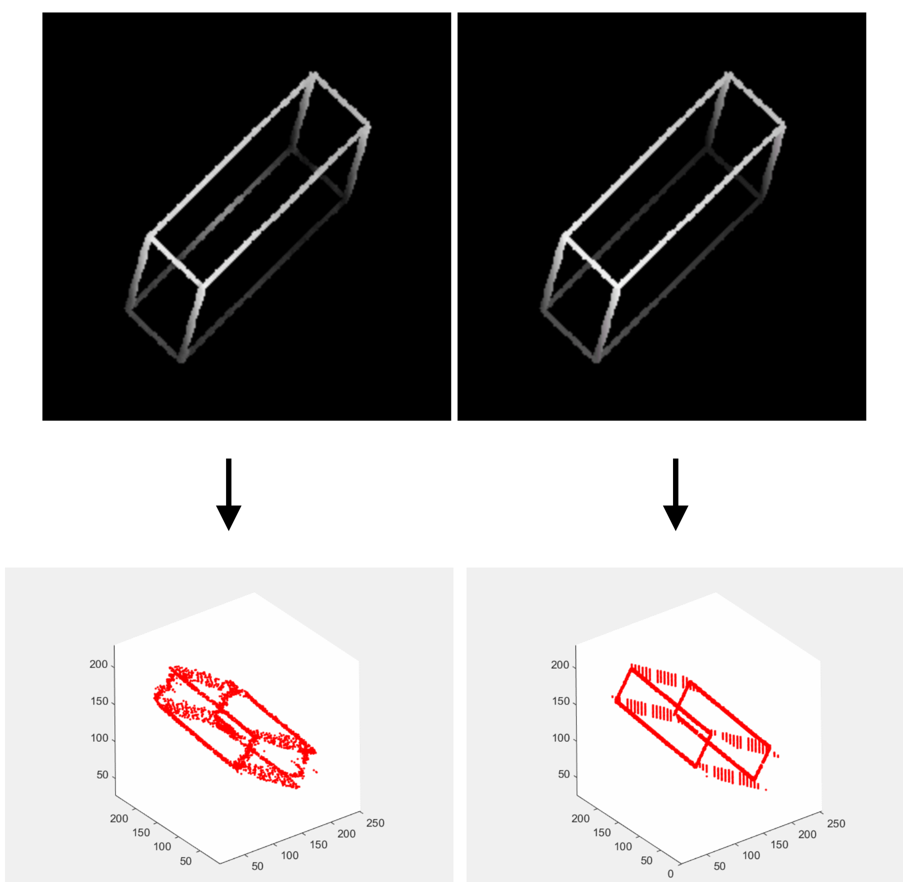


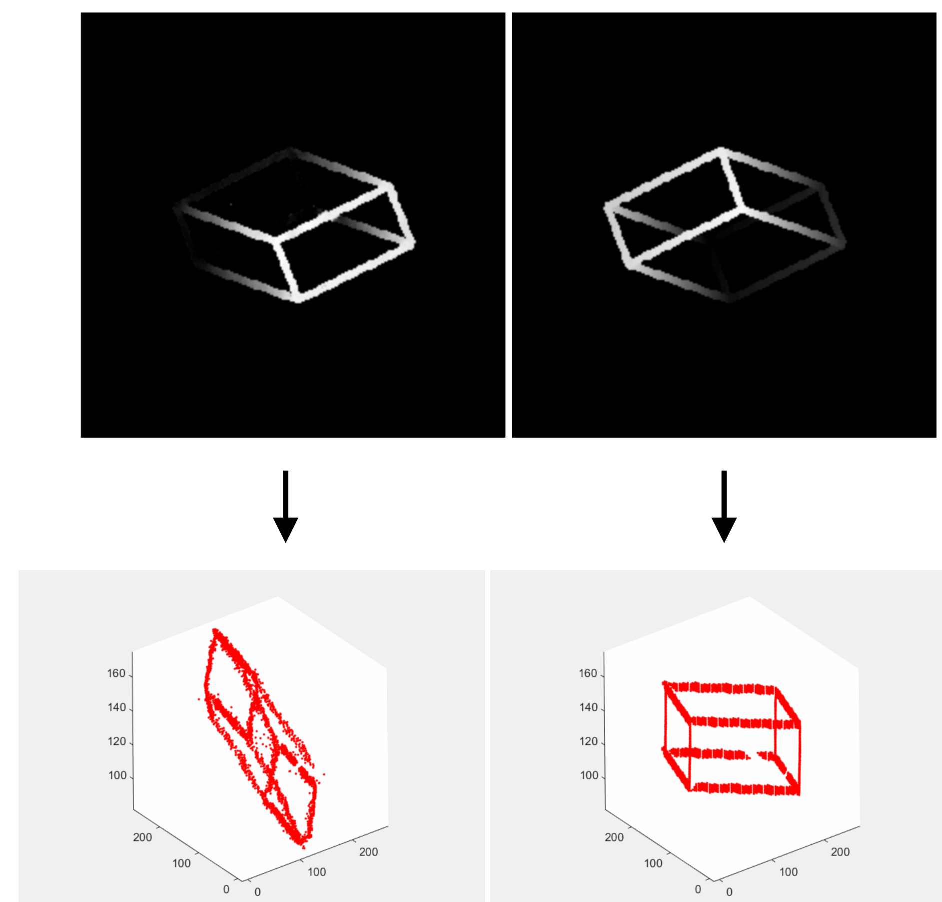


Output **Target**

Output **Target**

Depth:

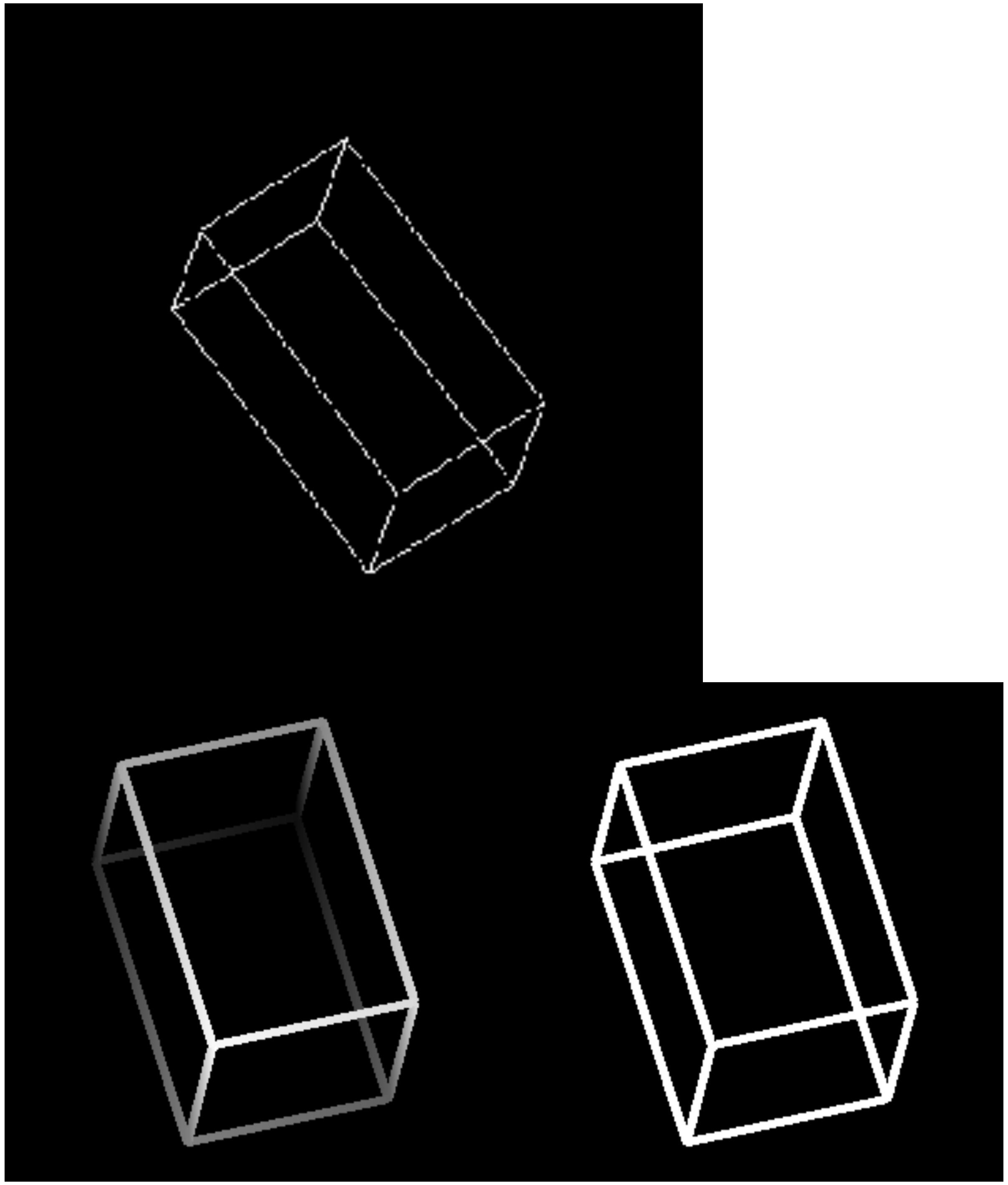




3D shape:

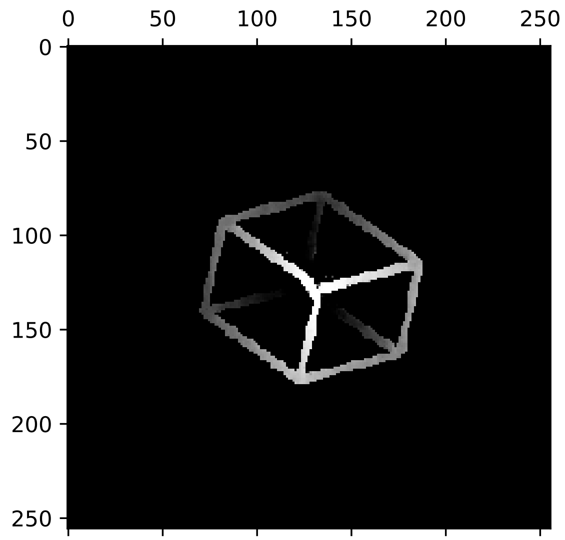
Data Generation

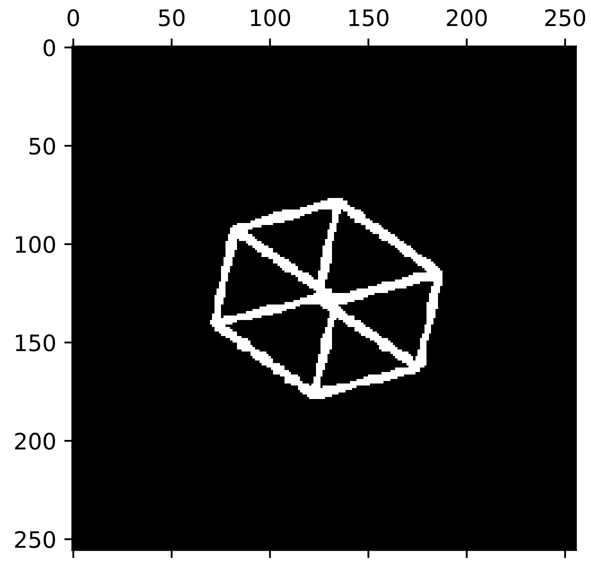
- Two Version:
- 5000 (In use of training) and 20000 (Generating, in improvement)
- “Shape picture” is 299*299 logic image,”Depth picture” is 299*299 grey image.
- Several Issues:
    - Direction
    - Edge width
    - Crossing point

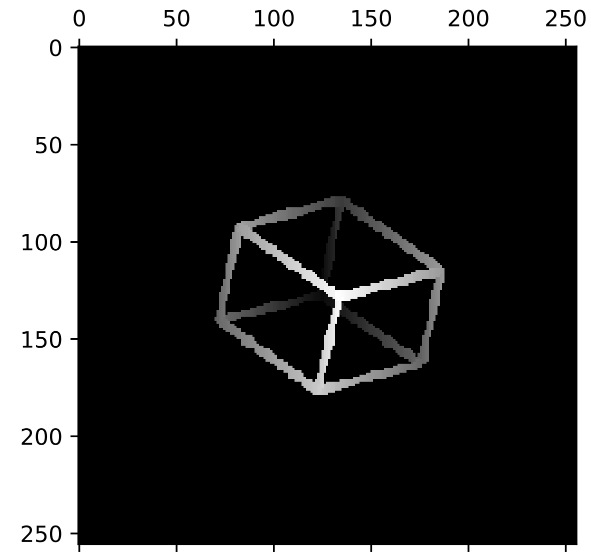


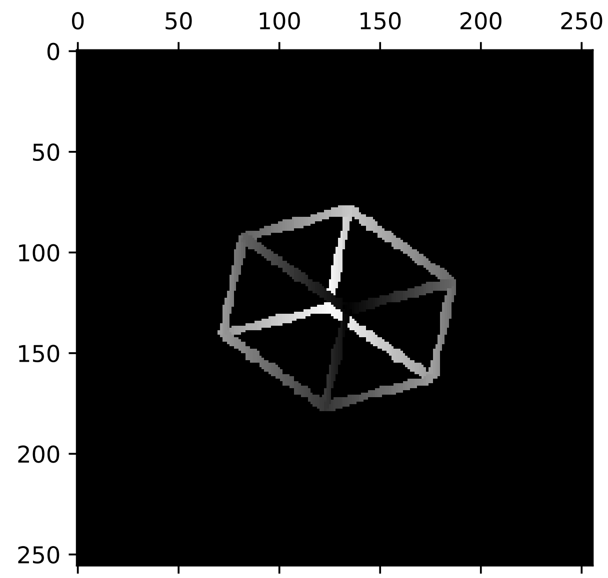
Evaluation Method

- Mask output data with non-zero pixels in test data
- $Error = \sum_{masked\ pixel} (\frac{G_{test} - G_{output}}{n_{pixel \neq 0}})^2$
- Minimum of Original and Inverse

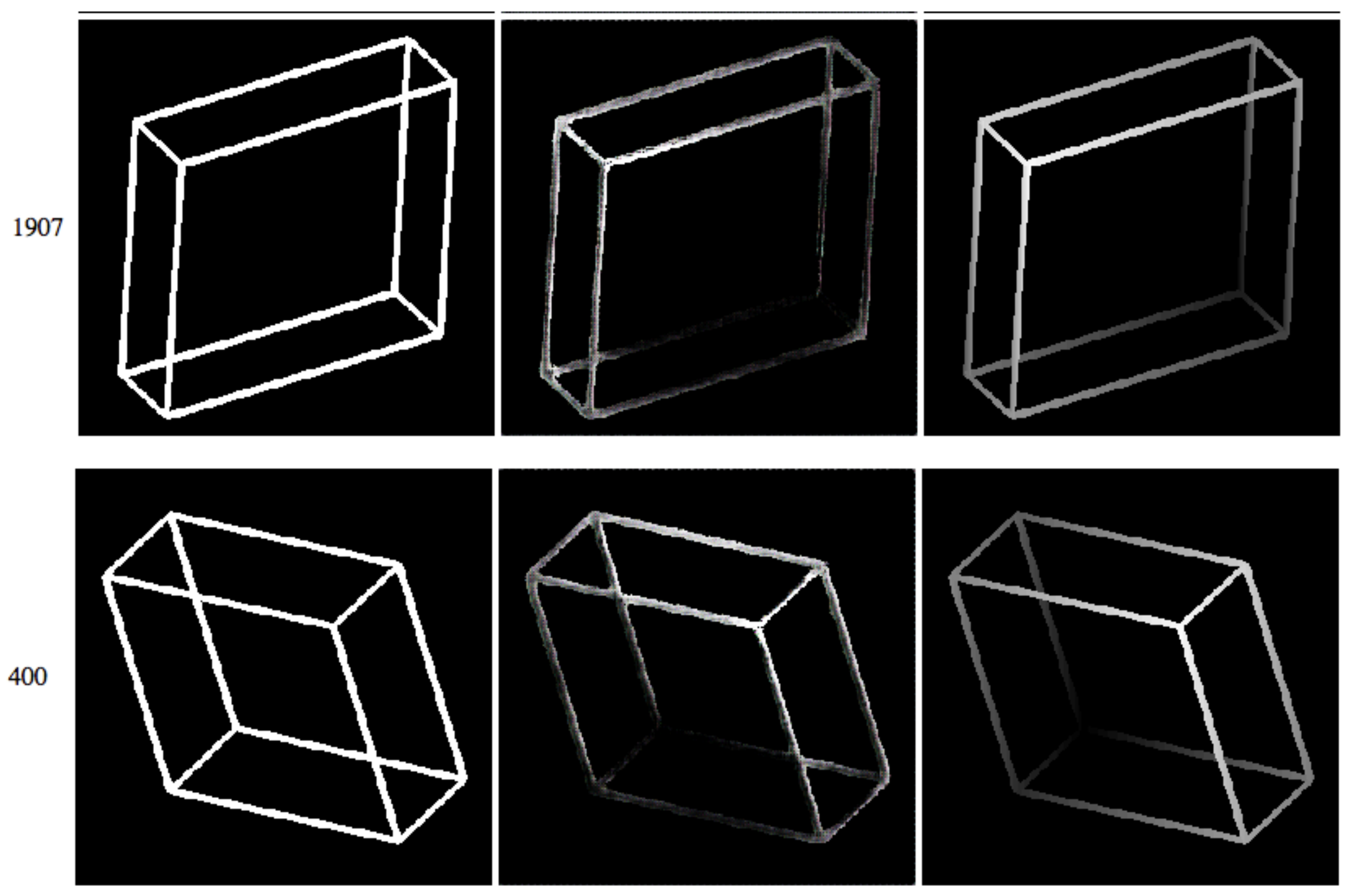








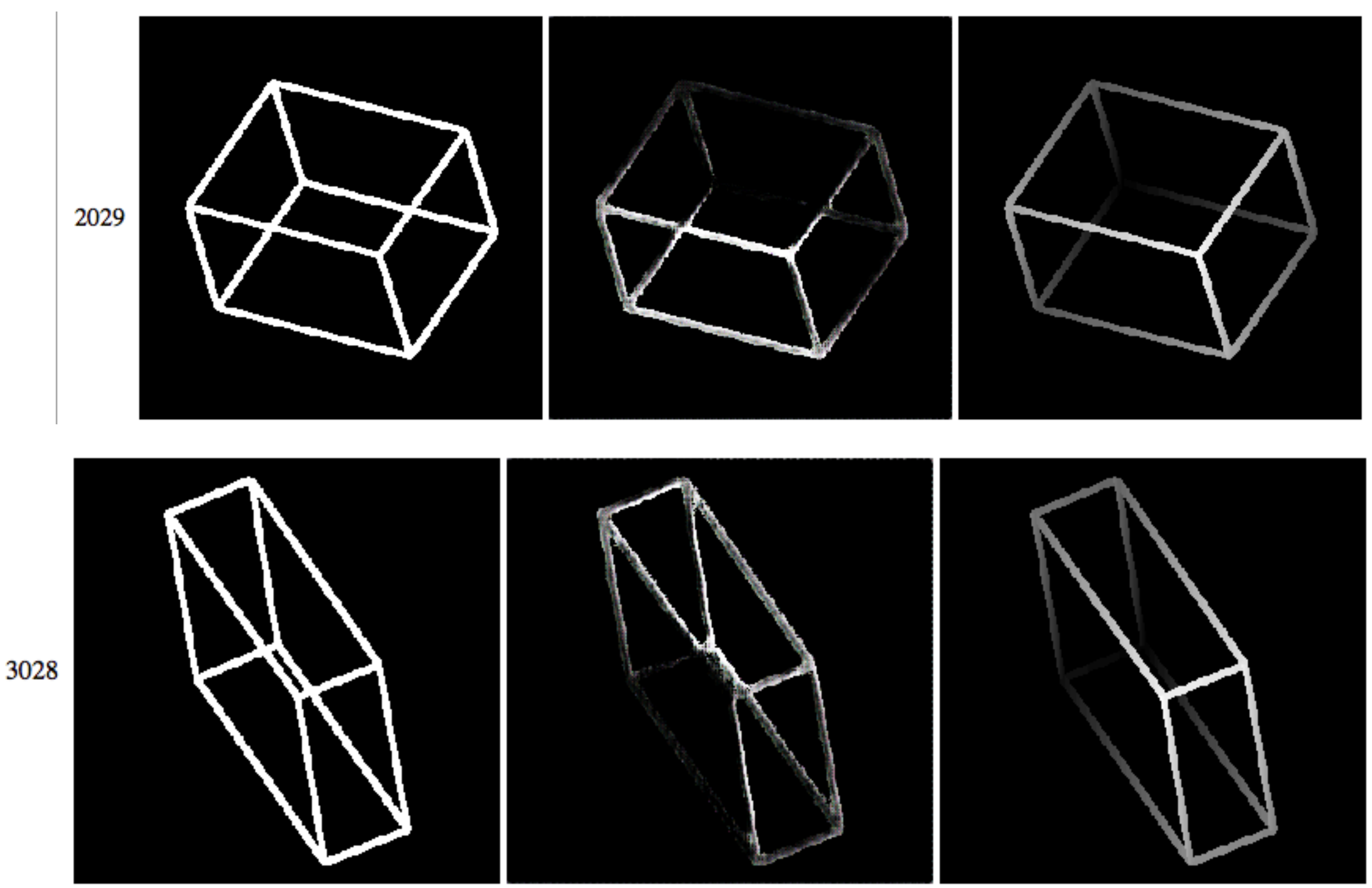
E < 1 (2% of test output)



$E = 0.56$

$E = 0.97$

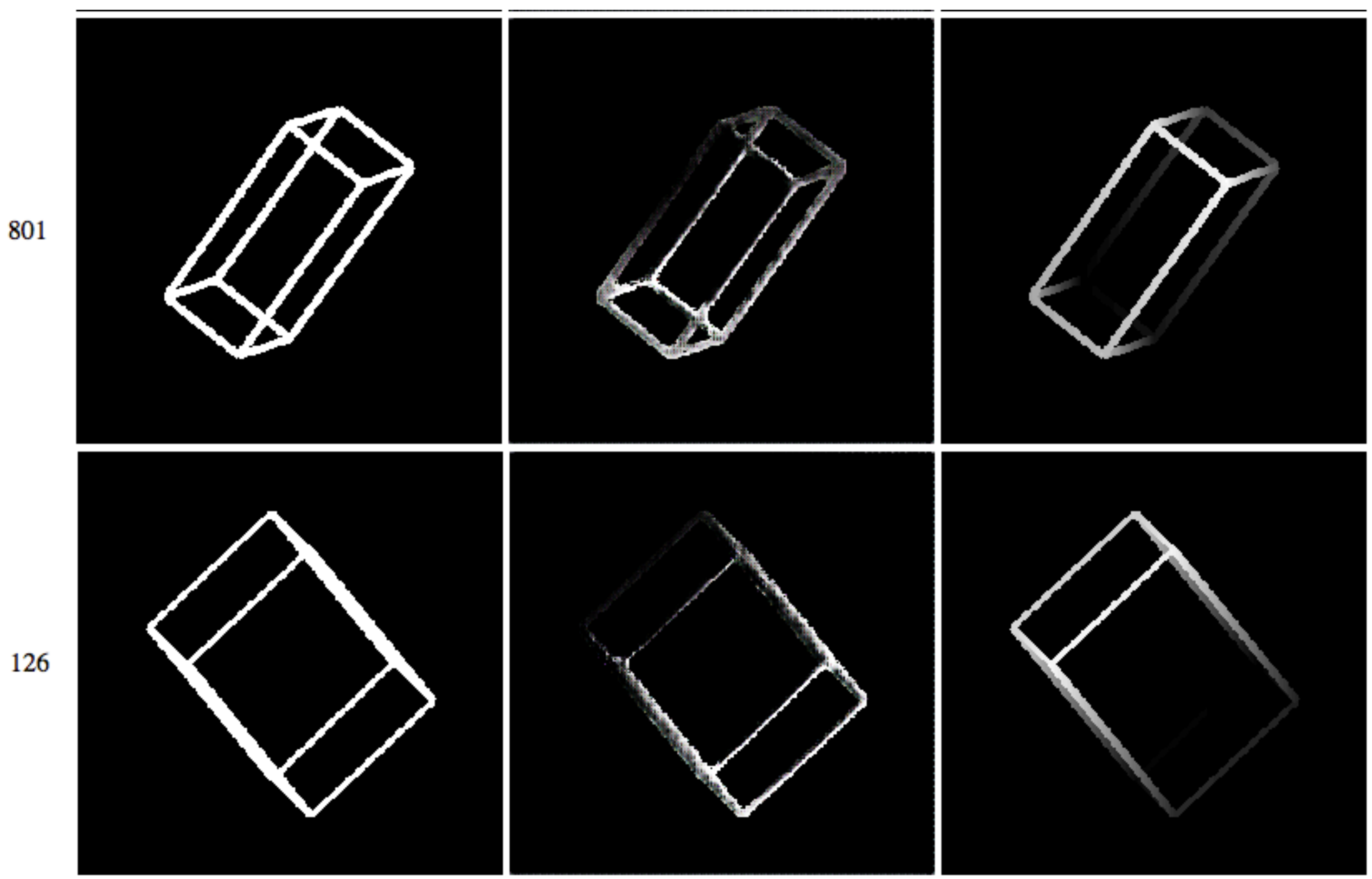
E < 2 (24% of test output)



$$E = 1.53$$

$$E = 1.99$$

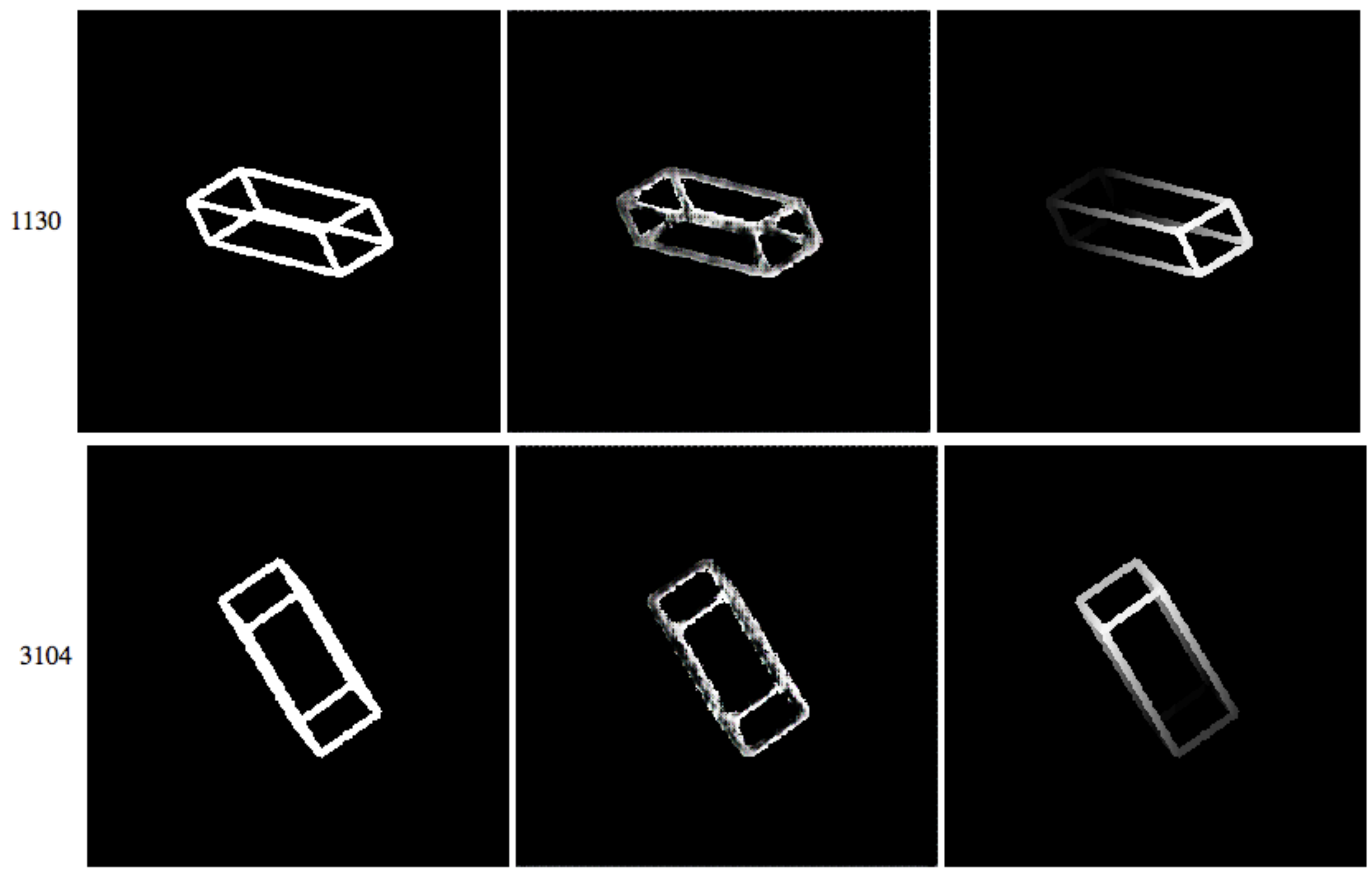
E < 3 (60% of test output)



$$E = 2.66$$

$$E = 2.97$$

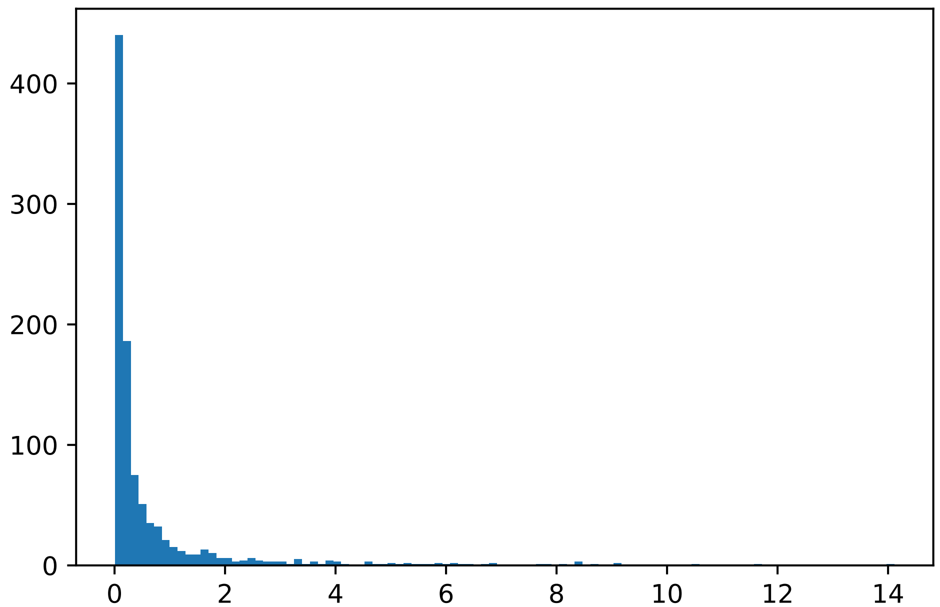
E < 5 (87% of test output)



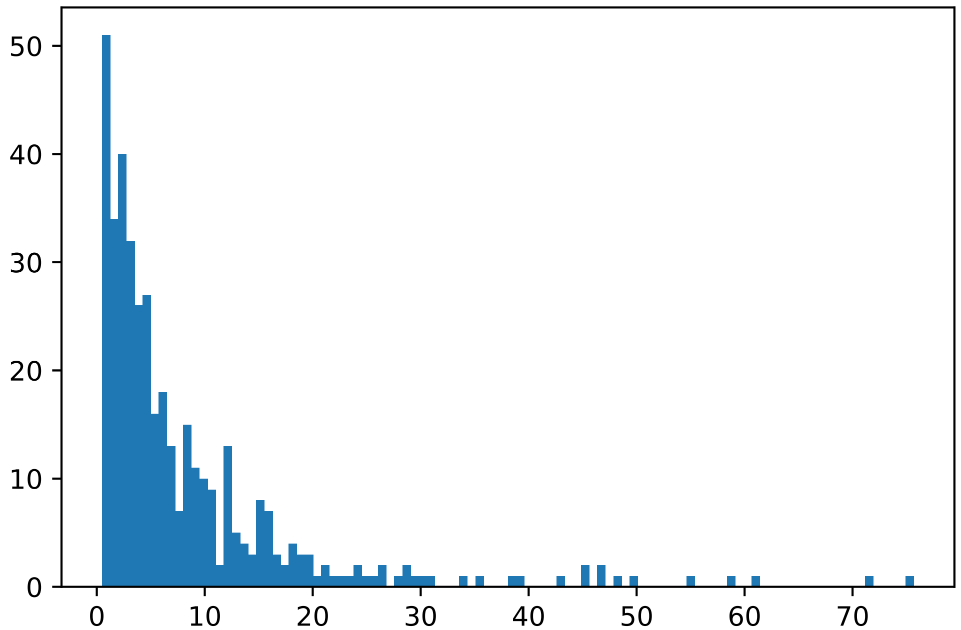
$$E = 4.92$$

$$E = 4.98$$

histogram



400
300
200
100
0
0
2
4
6
8
10
12
14



50
40
30
20
10
0
0
10
20
30
40
50
60
70

Future Works

- Regenerate only big images
- Smooth
- More data
- Other Approaches:
  - Regression
  - Laplacian