HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

# AUTOTOON

## Automatic Geometric Warping for Face Cartoon Generation

Nguyen Thi Hoang Linh - 20224324
Le Ngoc Huong - 20224315
Class: 152446

ONE LOVE. ONE FUTURE.

# Outline

Introduction

Dataset

Model Architecture

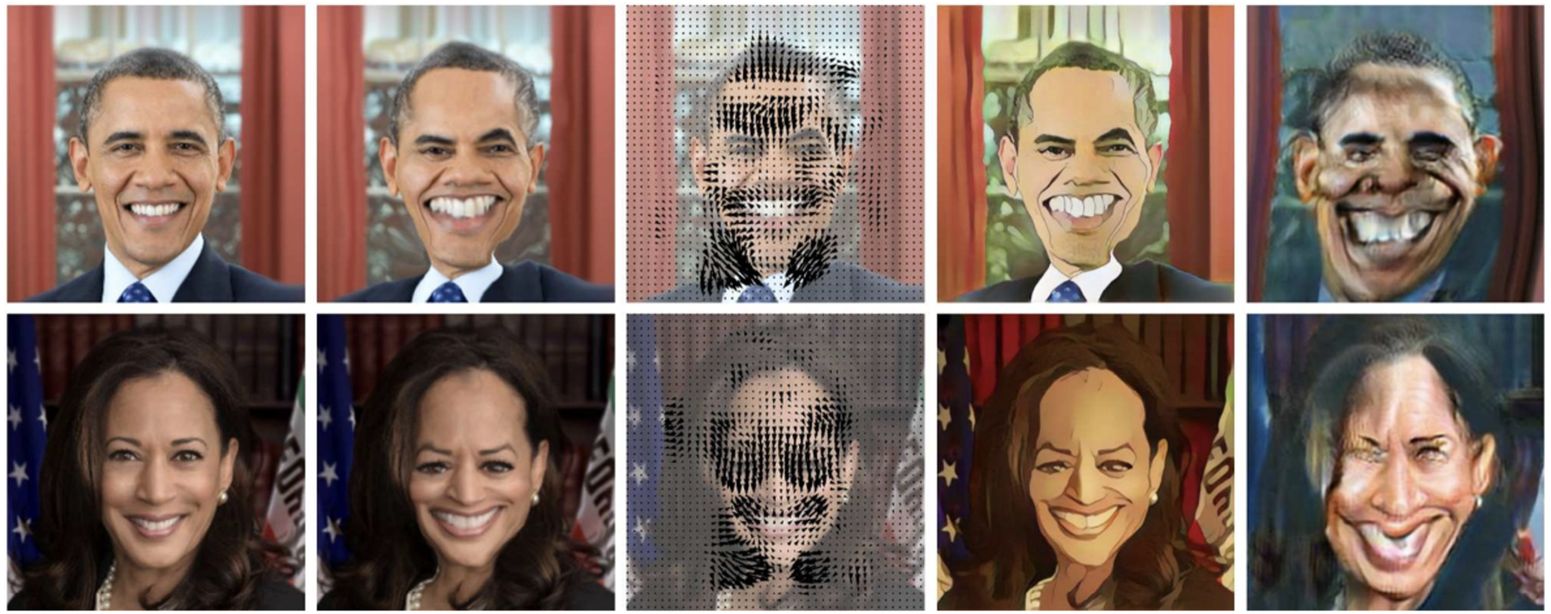
Progress

Evaluation

New Idea

Conclusion

# Introduction



(a) Input Photo (b) Cartoon (Ours) (c) Warping Field (Ours) (d) Caricature (Ours + CartoonGAN [6]) (e) Caricature (WarpGAN [28])

Figure 1: Example images

**About this project:**
Caricature: artists use creative exaggeration to accentuate unique facial traits
Input: a standard portrait photo
Output: an exaggerated cartoon version

**Target: an automated, end-to-end pipeline**
Step 1: Adding a geometric warp to the face to exaggerate features
Step 2: Stylized the twisted image for an artistic impact (style transfer).



The figure is from [1]

# Dataset

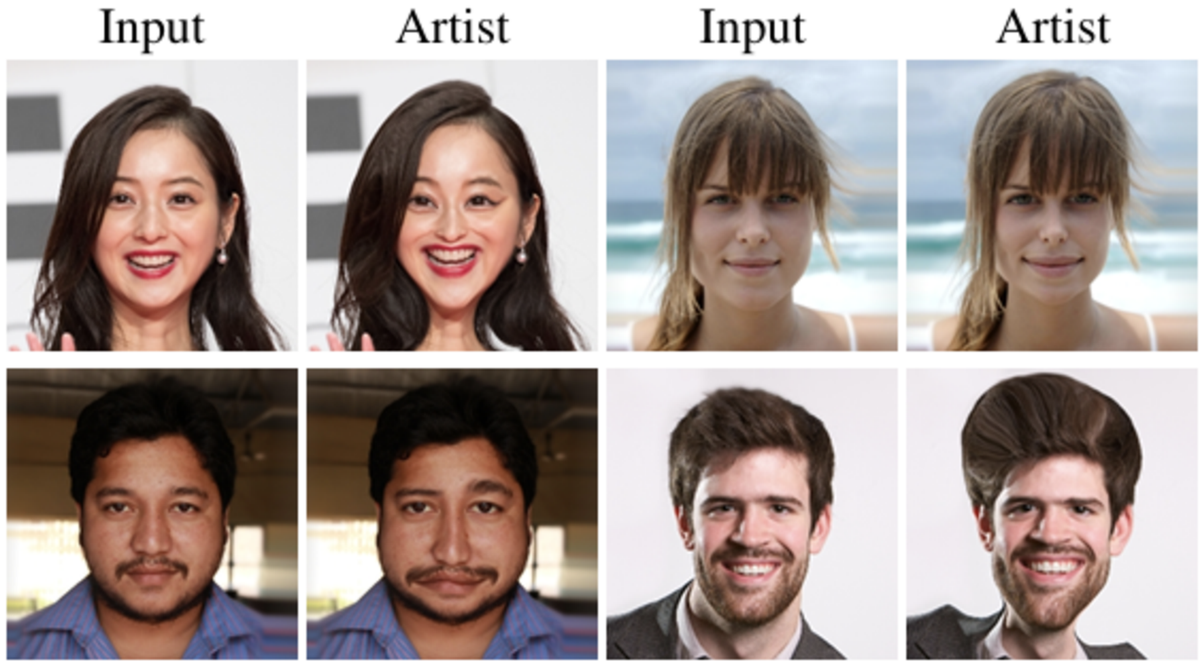



Figure 2: Dataset

101 frontal-facing portrait images will be collected from Flickr.

Ground truth: were warped via Adobe Photoshop by two caricature artists with similar styles.

The dataset includes 101 image pairs, with 90 for training and 11 for validation.

The test set, collected separately from public platforms, contains images without ground-truth labels.

The figure is from [1]

# Model Architecture

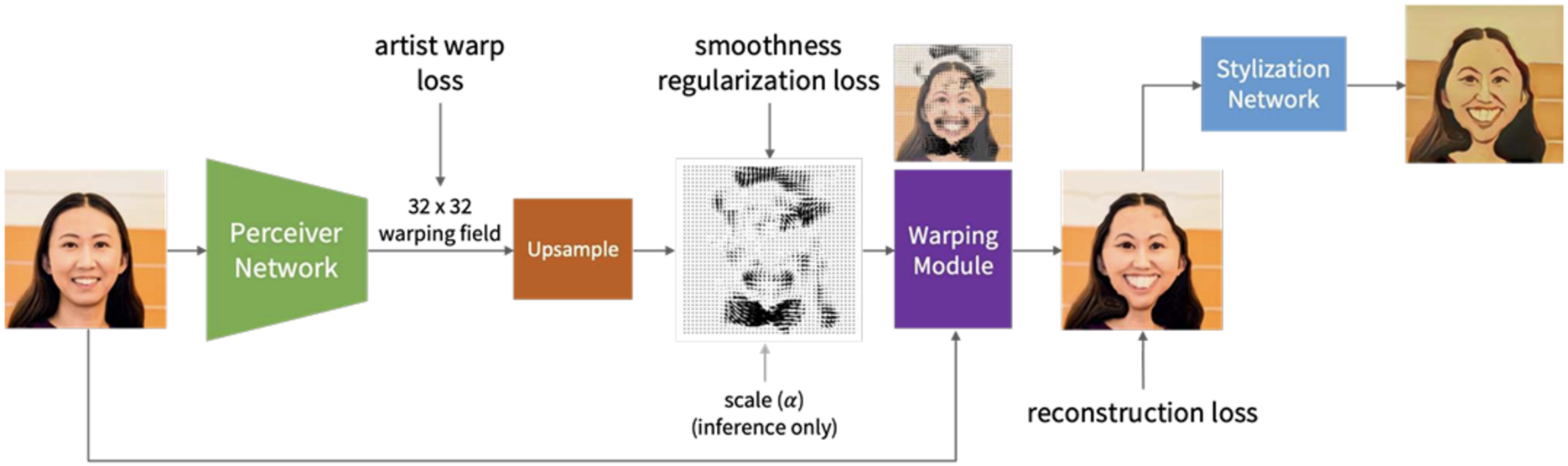



Figure 3: AutoToon model architecture and training losses.



The figure is from [1]

# Model Architecture

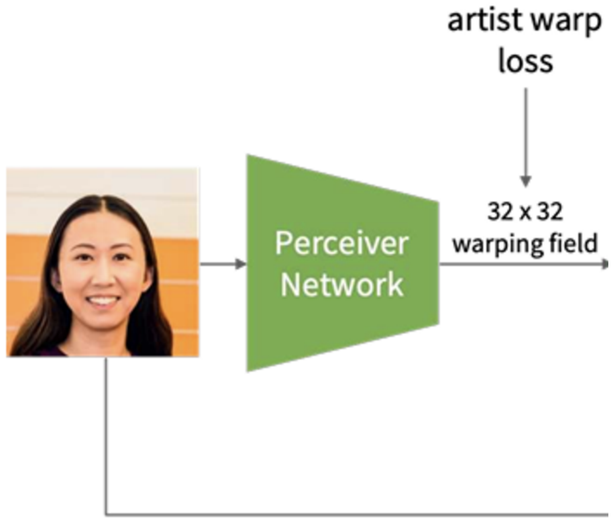



Figure 4: The Preceiver Network

## THE PRECEIVER NETWORK

A shortened version of the Squeeze-and-Excitation Network (SENet50) that has been pretrained using the VGGFace2 Dataset.

Retaining only the initial layers up to and including the second bottleneck block, followed by an adaptive average pooling layer with an output size of 32 × 32 × 2.

Truncating the network aims to decrease its capacity and prevent overfitting to the limited dataset.

Input: original image.
Output: the warping field, a flow field size HxWx2



The figure is from [1]

# Model Architecture

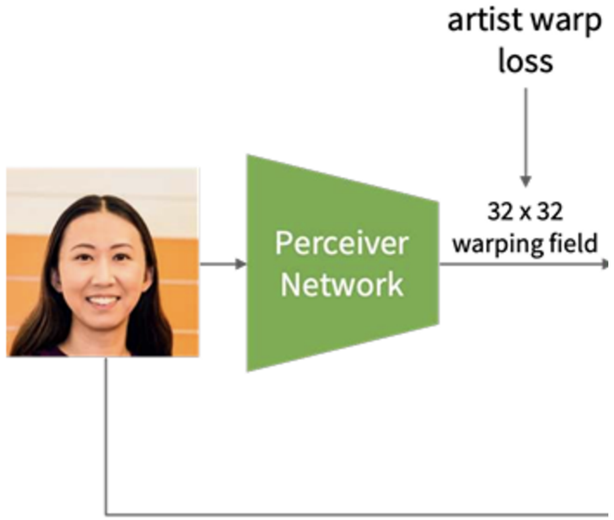



Figure 4: The Preceiver Network

## THE PRECEIVER NETWORK CODE

```
class AutoToonModel(nn.Module):
    def __init__(self, init_type='kaiming', models_root='./models',
                 lr=1e-4, lrd=1e-1, wd=0, force_cpu=False, train=True):

        self.upsample = nn.UpsamplingBilinear2d(size=256)

    def forward(self, x):
        flow = self.senet(x) / 100  # normalize to well within [-1, 1] range
        flow_norm = self.upsample(flow)
        warped = self.flow_warp(x, flow_norm)
        return warped, flow_norm, flow
```



The figure is from [1]

# Model Architecture

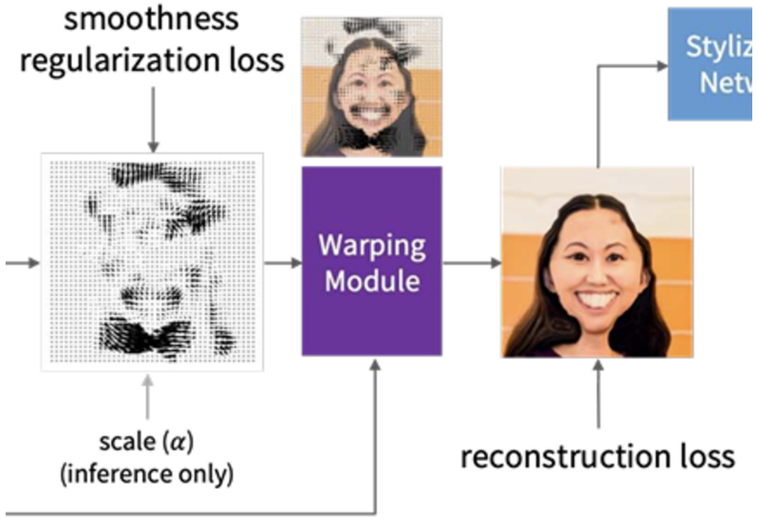



Figure 5: Warping Module

## WARPING MODULE

Applies the warping field to the image.

Input: smoothed warping field and original image
Output: warped image



The figure is from [1]

# Model Architecture

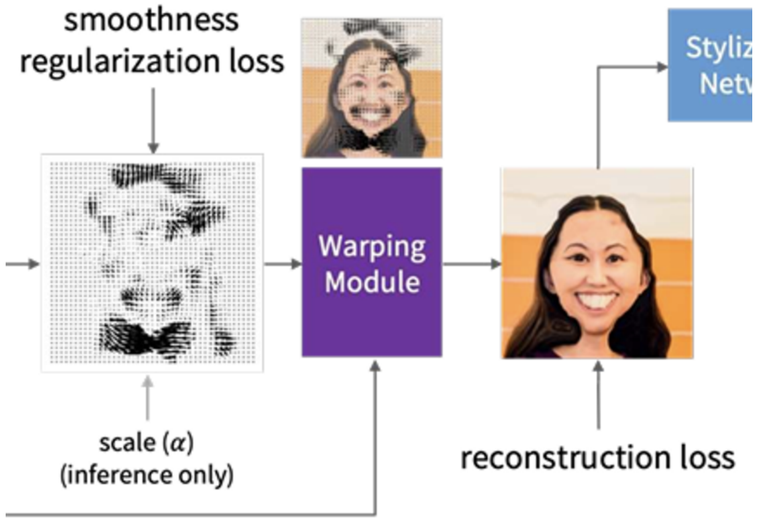



Figure 5: Warping Module

## WARPING MODULE CODE

```python
def flow_warp(self, x, flow, padding_mode='border'):
    # Ensure that the spatial dimensions of `x` and `flow` match
    assert x.size()[-2:] == flow.size()[-2:]
    # Extract batch size (n), channels (c), height (h), and width (w) from `x`
    n, _, h, w = x.size()
    # Create x-coordinates ranging from 0 to w-1, expanded to size (h, w)
    x_ = torch.arange(w).view(1, -1).expand(h, -1)
    # Create y-coordinates ranging from 0 to h-1, expanded to size (h, w)
    y_ = torch.arange(h).view(-1, 1).expand(-1, w)
    # Stack x and y to form a grid of coordinates (2, h, w)
    grid = torch.stack([x_, y_], dim=0).float().to(self.device)
    # Add batch dimension and replicate the grid for all batches (n, 2, h, w)
    grid = grid.unsqueeze(0).expand(n, -1, -1, -1).clone()
    # Normalize x-coordinates from [0, w-1] to [-1, 1]
    grid[:, 0, :, :] = 2 * grid[:, 0, :, :] / (w - 1) - 1
    # Normalize y-coordinates from [0, h-1] to [-1, 1]
    grid[:, 1, :, :] = 2 * grid[:, 1, :, :] / (h - 1) - 1
    # Add the optical flow to create a warped grid
    grid = grid + flow
    # Rearrange dimensions from (n, 2, h, w) to (n, h, w, 2) for `F.grid_sample`
    grid = grid.permute(0, 2, 3, 1)
    # Move `x` to the same device as the grid (CPU or GPU)
    x = x.to(self.device)
    # Warp the input `x` using the computed grid, with the specified padding mode
    return F.grid_sample(x, grid, padding_mode=padding_mode)
```



The figure is from [1]

# Model Architecture

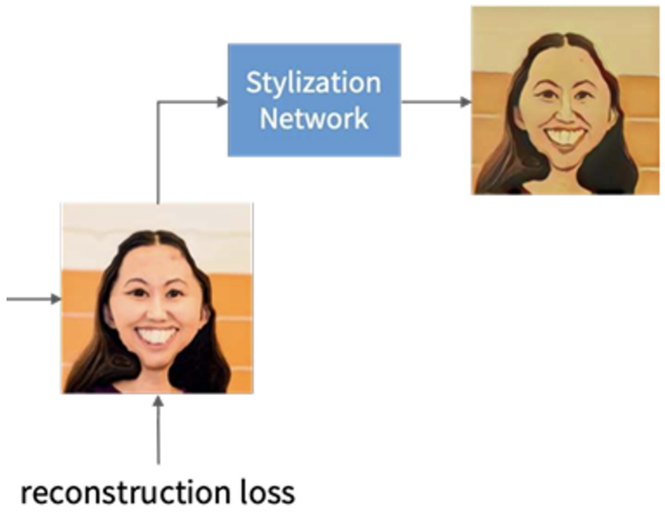



Figure 6: Stylization Network

## STYLIZATION NETWORK

Applying style transfer to the warped image for an artistic effect.

Using a pretrained model: CartoonGAN

Input: warped image
Output: final image

The figure is from [1]

# Progress

## PREPROCESS

Handle gray scale image.

Augmented data: random crop, flip, resize.

## MODEL DEVELOPMENT

Investigating suggested models, which include SENet50

## TRAINING AND OPTIMIZATION

Training: Train the model using the preprocessed data with appropriate hyperparameters.

Validation: Evaluate the model's performance on a validation dataset.

Optimization: Fine-tune model architecture or hyperparameters for better results.

# Evaluation

## HUMAN EVALUATION

Good

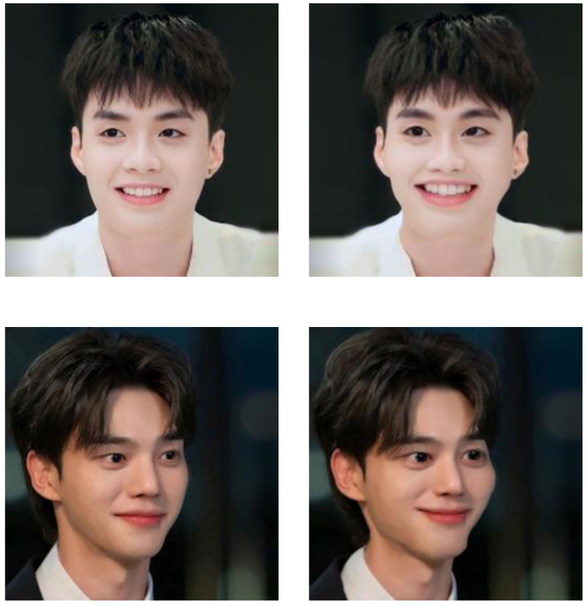


Figure 7: Good Results

Bad

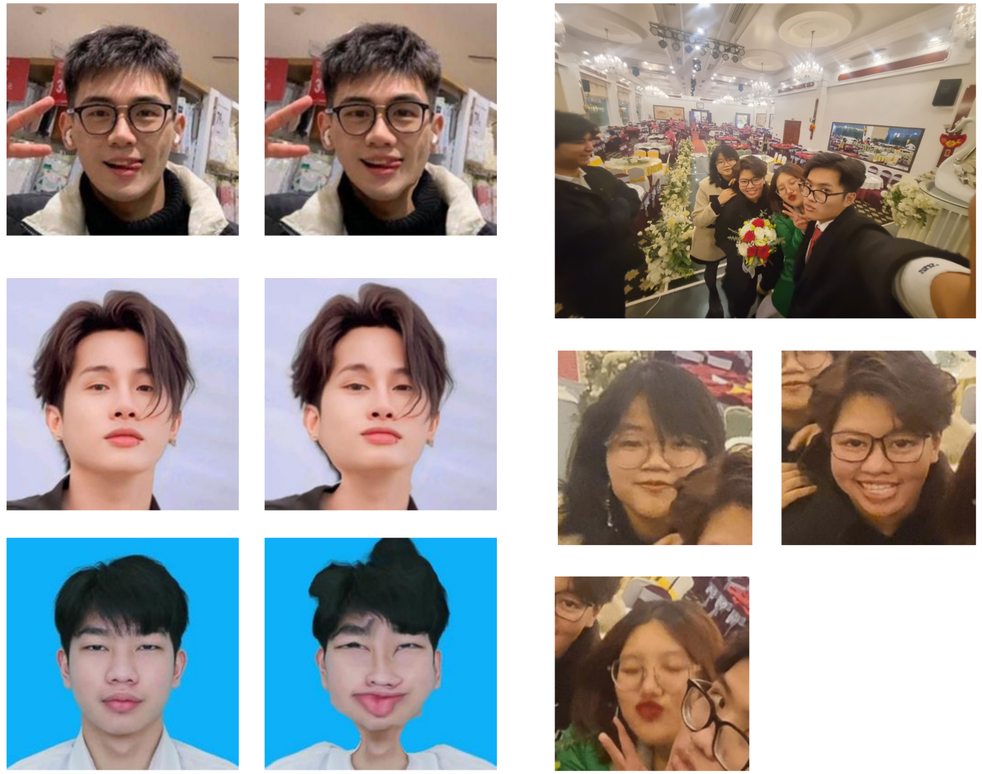


Figure 8: Bad Results

# Evaluation

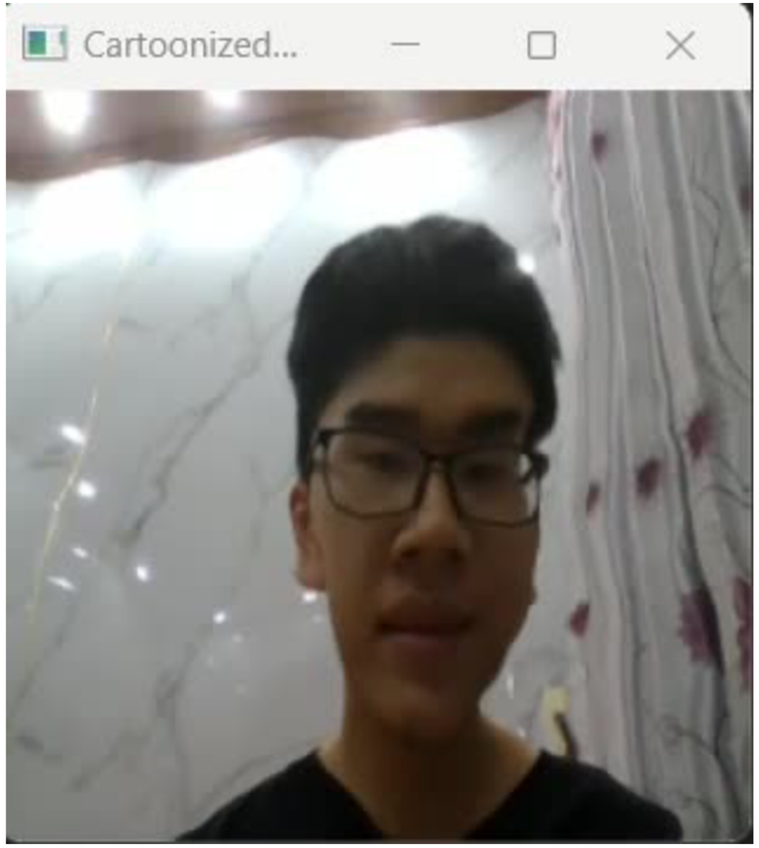
## METRICS

MSE (Mean Squared Error): 4.73
-> Quite small relative to the range of pixel values, meaning there are no extreme discrepancies between the ground truth and the output images.

SSIM (Structural Similarity Index): 0.982
-> Excellent, the model is able to preserve the structural features of the images very well, with minimal perceptual differences.

PSNR (Peak Signal-to-Noise Ratio): 40.58 dB
-> Very good, the output image is of high quality and very similar to the ground truth, with minimal distortion or noise.

# New Idea




## CARTOON WEBCAM FEED

The Cartoon Webcam Feed is a real-time application that processes live video feed from a webcam and applies a cartoon-style transformation to each frame.

Use OpenCV for webcam feed handling.

Features: Real-Time Processing; Deep Learning-Based Cartoonization and User Interaction.

# Conclusion

AutoToon offers flexibility and detail, outperforming state-of-the-art methods in exaggerating facial features. AutoToon is highlighted to be a potential project in addressing real-world applications.

However, while this system shows good results in caricature exaggeration, it still faces some challenges.

The future work can explore refining warping smoothness, improving identity preservation, and adapting to diverse artist styles through few-shot learning.

# Reference

[1] Julia Gong, Yannick Hold-Geoffroy, and Jingwan Lu. AutoToon: Automatic Geometric Warping for Face Cartoon Generation, 2020.

HUST

hust.edu.vn fb.com/dhbkhn

# THANK YOU !