Hello!

Welcome everyone :)

Student: Johnny Borkhoche
Goal: Introduction to my bachelor semester project

Background Information

- I am a 3rd year Bachelors student @ EPFL IC section, previously was in EL section
- Level: Bachelor semester project
- Supervisors:
  - Prof. Dr. Bourlard Hervé
  - Mr. Sylvain Calinon
  - Tobias Löw

Learning of non-photorealistic renderings for a caricaturist robot

- Initial Description: This project aims to review, categorize and test existing approaches in deep learning to map an image to a set of trajectories that a robot can then draw on a canvas, by taking the example of portrait caricatures as an example of generative non-photorealistic rendering (NPR) problem (see CariGAN or WarpGAN as starting examples). Several data processing pipelines can be considered, with parts that can be specified manually, and others that can be learned from examples. It also includes the potential consideration of intermediary steps, such as detecting the location of facial features (e.g., by using Google's MediaPipe Face Mesh), image-to-image processing applying pencil/brush sketch rendering effects, or applying a distortion mask for a warping transformation defined explicitly. The project also aims at studying metrics or methods to compare these different approaches. The selected approach(es) will take into account the amount of data that these methods require, the availability of these data and of pretrained models, and the possibility of employing the selected method(s) on smartphones or CPUs (instead of GPUs).

Learning of non-photorealistic renderings for a caricaturist robot

- Initial Description: This project aims to review, categorize and test existing approaches in deep learning to map an image to a set of trajectories that a robot can then draw on a canvas, by taking the example of portrait caricatures [illegible] ering (NPR) problem (see CariGAN or WarpGAN [illegible] elines can be considered, with parts that can be [illegible] om examples. It also includes the potential [illegible] g the location of facial features (e.g., by usin [illegible] rocessing applying pencil/brush sketch re [illegible] arping transformation defined explicitly. The [illegible] ompare these different approaches. The sele [illegible] f data that these methods require, the availability of these data and of pretrained models, and the possibility of employing the selected method(s) on smartphones or CPUs (instead of GPUs).

Learning of non-photorealistic renderings for a caricaturist robot

- [Shortened] Description:

  This project aims to **realistically warp and caricaturize** any **portrait** image. Said image will then be mapped to a set of trajectories that a robot can draw on a canvas.

Learning of non-photorealistic renderings for a caricaturist robot

- [Shortened] Description:

  This project aims to **realistically warp and caricaturize** any **portrait** image. Said image will then be mapped to a set of trajectories that a robot can draw on a canvas.

- My role in all of this was progressively defined through the first few weeks of literature review and feedback

What I am trying to achieve

- My main goal is to find a way to caricaturize human faces as realistically as I possibly can

What I am trying to achieve

- My main goal is to find a way to caricaturize human faces as realistically as I possibly can
- Some approaches I considered:
  - Training a deep network
  - Image processing
  - Affine transformations

What I am trying to achieve

- My main goal is to find a way to caricaturize human faces as realistically as I possibly can
- Some approaches I considered:
  - Training a deep network
  - Image processing
  - Affine transformations
- To do that, I did some of the following:
  - Reading papers
  - Finding interesting libraries
  - Brainstorming

What I started with

- State of the art regarding facial detection techniques
  - talk about project feedback 1.md if there’s time

Most Similar Works

- CariGAN
  - Generative Adversarial Network (GAN) for unpaired photo-to-caricature translation, which are called "CariGANs"

Most Similar Works

- CariGAN
  - Generative Adversarial Network (GAN) for unpaired photo-to-caricature translation, which are called "CariGANs"
  - Done in 2 main steps
    - CariGeoGAN: models the geometry-to-geometry transformation from face photos to caricatures
    - CariStyleGAN: transfers the style appearance from caricatures to face photos without any geometry deformation

Most Similar Works

- WarpGAN
  - a fully automatic network that can generate caricatures given an input face photo

Observations

- Both CariGAN and WarpGAN have impressive looking results
- Both methods are implementations of Generative Adversarial Networks which is a network widely used for image, video and voice generation

Observations

- Both CariGAN and WarpGAN have impressive looking results
- Both methods are implementations of Generative Adversarial Networks which is a network widely used for image, video and voice generation

- A flagrant issue, however, is that the outcomes are cartoon-like
- Not what we’re looking for

A different approach

- While Deep Learning architectures are appealing and useful, I had to scratch them out

A different approach

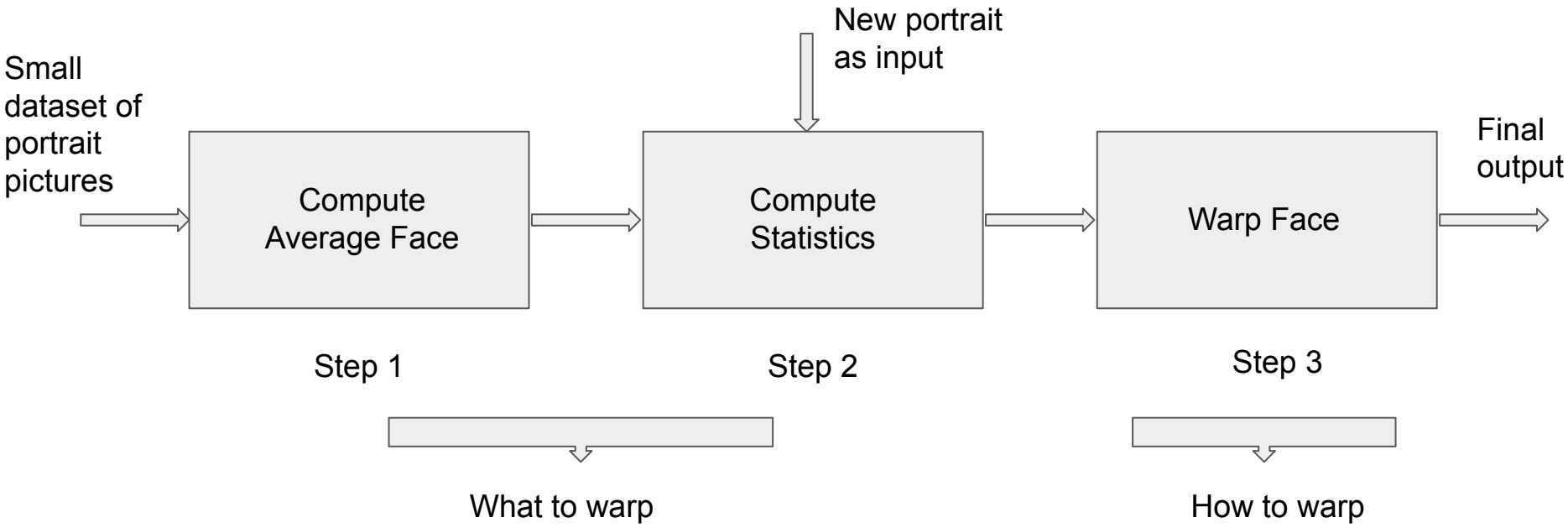
- While Deep Learning architectures are appealing and useful, I had to scratch them out
- Challenges:
  - Finding well annotated portrait datasets
  - Never built a DL model from scratch
  - Not necessary?

A different approach

- While Deep Learning architectures are appealing and useful, I had to scratch them out
- Challenges:
  - Finding well annotated portrait datasets
  - Never built a DL model from scratch
  - Not necessary?

- My answer was:
  - Computer Vision (openCV)
  - Trained ML algorithms (dlib)

Current Pipeline



Average Face - a little history

- Little historical fact: face averaging started with Francis Galton (Charles Darwin's cousin) back in 1878 when he came up with a new photographic technique for compositing faces by aligning the eyes. He wanted to catch more criminals!
- He noted the average face was always more attractive than the faces it was the average of.
- In fact, several researchers in the 1990s showed that people find facial averages much more attractive than individual faces. In one amusing experiment researchers averaged the faces of 22 miss Germany finalists of 2002. People rated the average face to be much more attractive than every one of the 22 contestants, including miss Berlin who won the competition.

Average Face - a little history

Real Miss Germany 2002

Virtual Miss Germany 2002

What do you think?

Average Face - a little history

Real Miss Germany 2002

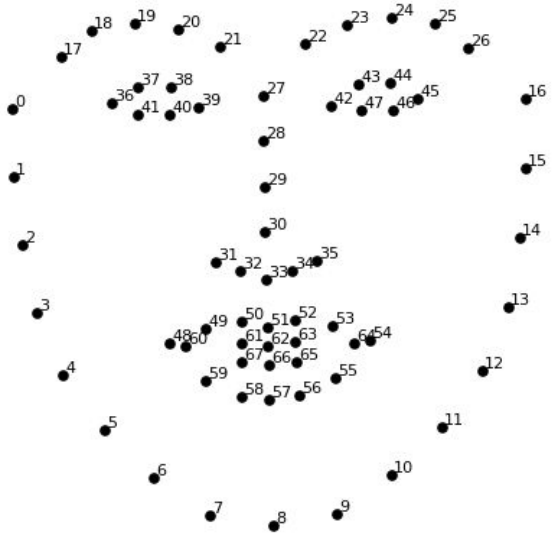
Virtual Miss Germany 2002

- An average face is also symmetric because the variations in the left side and the right side of the face are averaged out.

Average Face computation

Step 1: Facial feature detection

- For each image, we calculate 68 facial landmarks using dlib’s pre-trained facial recognition model.



Average Face computation

Step 2: Coordinate transformation

- Input images can vary in dimensions -> must bring them to same reference frame

Average Face computation

Step 2: Coordinate transformation

- Input images can vary in dimensions -> must bring them to same reference frame
- Idea:
  - Warp the faces to a 600x600 image such that the left eye corner is at pixel location (180, 200) and the right eye corner is at (420, 200). The result is what I call the output coordinate system

Average Face computation

Step 2: Coordinate transformation

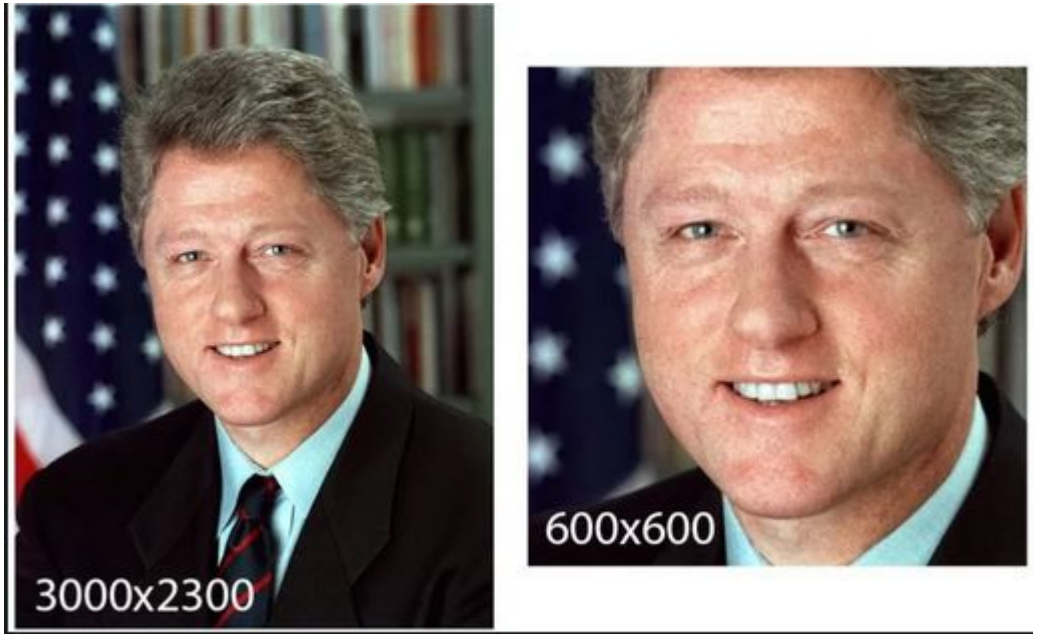
- Input images can vary in dimensions -> must bring them to same reference frame
- Idea:
  - Warp the faces to a 600x600 image such that the left eye corner is at pixel location (180, 200) and the right eye corner is at (420, 200). The result is what I call the output coordinate system
  - Explanation of choice: to make sure that the points were on a horizontal line and that the face was centered at about ⅓ of the height from the top of the image, I chose the left and right eye corners to respectively be at (0.3 x width, height / 3) and (0.7 x width , height / 3).

Average Face computation

Step 2: Coordinate transformation

- Input images can vary in dimensions -> must bring them to same reference frame
- Idea:
  - Warp the faces to a 600x600 image such that the left eye corner is at pixel location (180, 200) and the right eye corner is at (420, 200). The result is what I call the output coordinate system
  - Explanation of choice: to make sure that the points were on a horizontal line and that the face was centered at about ⅓ of the height from the top of the image, I chose the left and right eye corners to respectively be at (0.3 x width, height / 3) and (0.7 x width , height / 3).
  - Since we also know the position of the corners of the eyes in the original image (they are landmarks 36 and 45), I thus calculate a similarity transform to take the input coordinate system to the output coordinate system

Average Face Computation



Coordinate
Transformation example

Average Face computation

Step 3: Face Alignment

- We previously transformed the input coordinate system into the output coordinate system.
- This results in same-size images with the eye corners aligned.
- Still can’t calculate the mean face; the rest of the facial features aren’t well aligned

Average Face computation

Step 3: Face Alignment

- We previously transformed the input coordinate system into the output coordinate system.
- This results in same-size images with the eye corners aligned.
- Still can’t calculate the mean face; the rest of the facial features aren’t well aligned
- Solution: use dlib’s 68 landmarks to divide the face into triangular regions and align them before averaging pixel values

Average Face computation

Step 3: Face Alignment - Delaunay Triangulation

- To calculate the average face where the features are aligned, we first calculate the average of all transformed landmarks in the output image coordinates (we average the (x, y) values of the transformed landmarks)

Average Face computation

Step 3: Face Alignment - Delaunay Triangulation

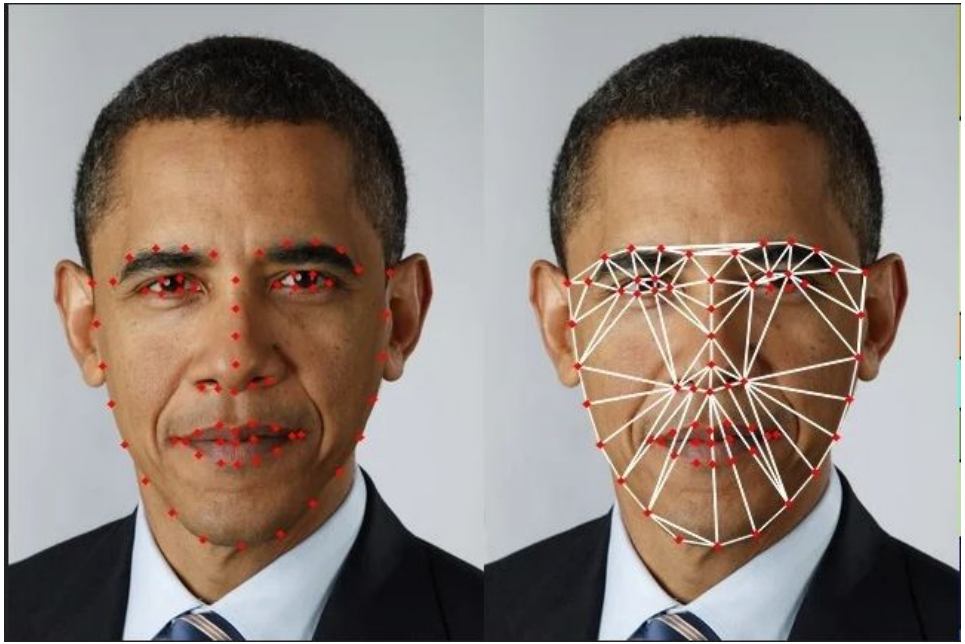
- To calculate the average face where the features are aligned, we first calculate the average of all transformed landmarks in the output image coordinates (we average the (x, y) values of the transformed landmarks)
- We add 8 boundary points to the output image to calculate à Delaunay Triangulation

Average Face computation

Step 3: Face Alignment - Delaunay Triangulation

- To calculate the average face where the features are aligned, we first calculate the average of all transformed landmarks in the output image coordinates (we average the (x, y) values of the transformed landmarks)
- We add 8 boundary points to the output image to calculate a Delaunay Triangulation
- This method breaks the image into triangles and results in a list of triangles represented by the indices of points in the 76 points (68 landmarks + 8 boundary)

Average Face computation



Example result of
Delaunay Triangulation

Average Face computation

Step 4: Warp Triangles

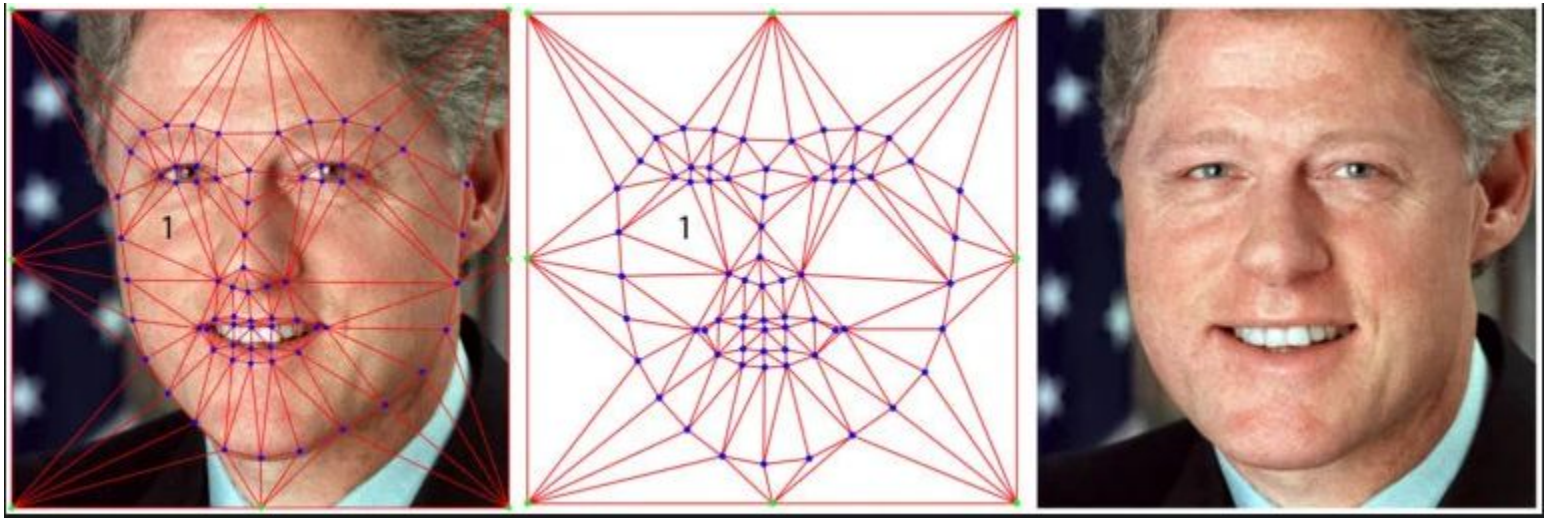
- Remember that we now possess a method to calculate Delaunay Triangulations
- We find the triangulation T1 of the transformed input image (i.e the output coordinate system found through the similarity transform), and T2 of the [not-yet-aligned] average facial landmarks that we've computed.

Average Face computation

Step 4: Warp Triangles

- Remember that we now possess a method to calculate Delaunay Triangulations
- We find the triangulation T1 of the transformed input image (i.e the output coordinate system found through the similarity transform), and T2 of the [not-yet-aligned] average facial landmarks that we've computed.
- We calculate an affine transformation from T1 to T2, then use it on all pixels of the un-aligned faces to finally obtain aligned faces

Average Face computation



Triangulation T1
of un-aligned face

Triangulation T2
of average face

Result of affine transform
on un-aligned face

Average Face computation

Final Step: Face averaging

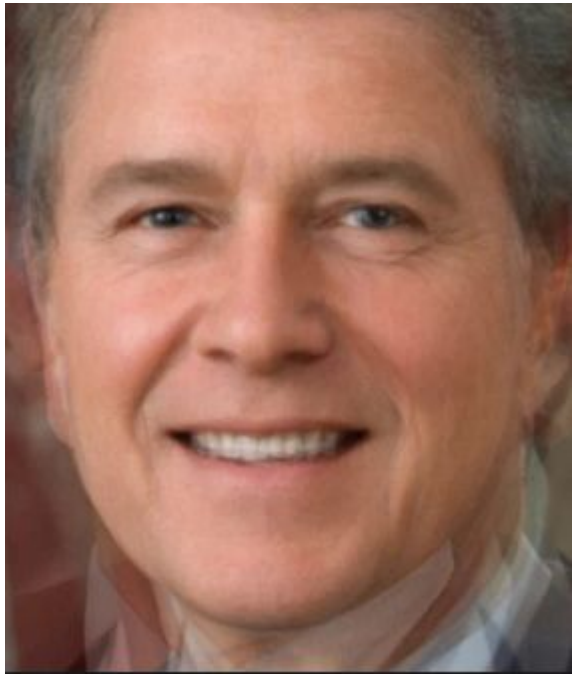
- When the previous step is applied to all input images (in the dataset), we obtain correctly warped faces proportional to the average landmark coordinates

Average Face computation

Final Step: Face averaging

- When the previous step is applied to all input images (in the dataset), we obtain correctly warped faces proportional to the average landmark coordinates
- We can now simply add the pixel intensities of all warped images and normalize them to obtain our final average face!

Average Face computation - Results



Average face resulting of 6 US presidents (from Carter to Obama)

Statistics Computation

- Where I currently am
- Aim is to calculate the distribution of some statistics from the average face computed in the previous step and find interesting data if possible
- Then use those statistics to compare with new portrait inputs and choose what part of them to caricaturize

Statistics Computation work in progress

Examples of Statistics

- Face Width
- Face Length
- Distance in-between the eyes
- Mouth width
- Mouth length
- etc

To be determined