Local warp and cross-dissolve

Group: PhotoshopThis
Haard Panchal 201501153
Saurabh Ravindranath 201501159
Eashwar Subramanian 201501163

Aim

The aim of the project is to morph two images such that we can see the transition of one image to the other over time.

For this, we first generate correspondence points for both the images to obtain the average shape. We do this by a process called local warping. We then find the average color by cross dissolving the warped images.

Warping

To achieve warping, we perform the following steps:

1. Generate correspondence points between the two images.
2. Use the in-built Delaunay triangulation function in matlab to obtain the triangulation of the corresponding points.
3. We then interpolate the points in the two images as a function of time, t (t = [0, 1]), using the following formula : $(1-t)*p_1+t*p_0$, where p_0 and p_1 are the corresponding points in the two images.

Warping

In order to obtain the positions of the remaining points in the images, we use the triangulation that we have computed, we perform the following steps:

1. We first determine which triangle the given point belongs in.
2. We then compute the barycentric-coordinates of the point using the following expression : $x = p_1{}^*a + p_2{}^*b + p_3{}^*c$, where x is the required point, a, b, c are the vertices of the triangle and p_1, p_2, p_3 are the coefficients. Here, $p_1 + p_2 + p_3 = 1$.
3. As there are 3 equations and 3 variables, the coordinates of the point in the new image can be computed.

Computing The Image Color

We compute the color of the different images across time t using the cross-dissolve method.

1. We first warp the 2 images.
2. We then compute the image at time t using the following expression : $Image_t = (1-t)*Image_1 + t*Image_2$

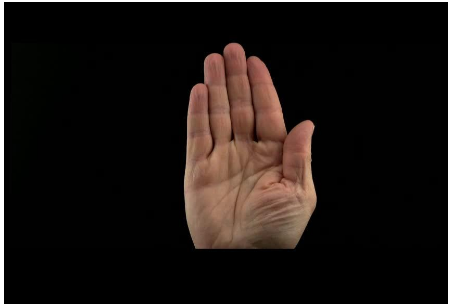
Extension - Application in Animation

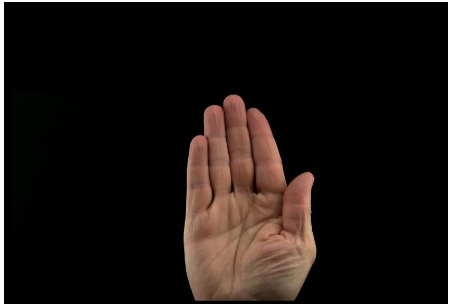
We have extended the image morphing idea so that given 2 images of the object in different locations, the program generates the intermediate frames necessary so that an animation is generated of the object in motion, from its position in image A to its position in image B.

This is done by choosing a set of points such that the image is contained within it, in both the images. So, as these points get interpolated over time, the positions of the triangles change uniformly, so that the position of the entire object get's changed along with it, due to the cross-dissolve algorithm used.

Results

The object we have chosen is a hand, which is in position A in image 1 and position B in image 2. Using our program, we marked points so that the hand is contained within the points.





These are the 2 images given as input to the program. The output of the program is the following gif:

https://drive.google.com/open?id=0B9D4yQ7vx6qVZWN6elhHdTRJSTA

Results

- We have a demonstration of the working of the program, as it morphs 8 different images sequentially, showing the evolution of the Ford Mustang models over time. We used 20 different input points to achieve our result, where images were generated at an interval of 0.025 sec: https://drive.google.com/open?id=0B9D4yQ7vx6qVY2FLT2NfQWliaEE

- We have a demonstration of the program morphing 2 football players: https://drive.google.com/file/d/0B48hZ3rGe8E0c2ZoaWJzY0M0UkE/view?usp=sharing

Style Transfer for Headshot Images

“Style Transfer for Headshot Portraits”

YiChang Shih, Sylvain Paris, Connelly Barnes, William T. Freeman, Fredo Durand

SIGGRAPH 2014

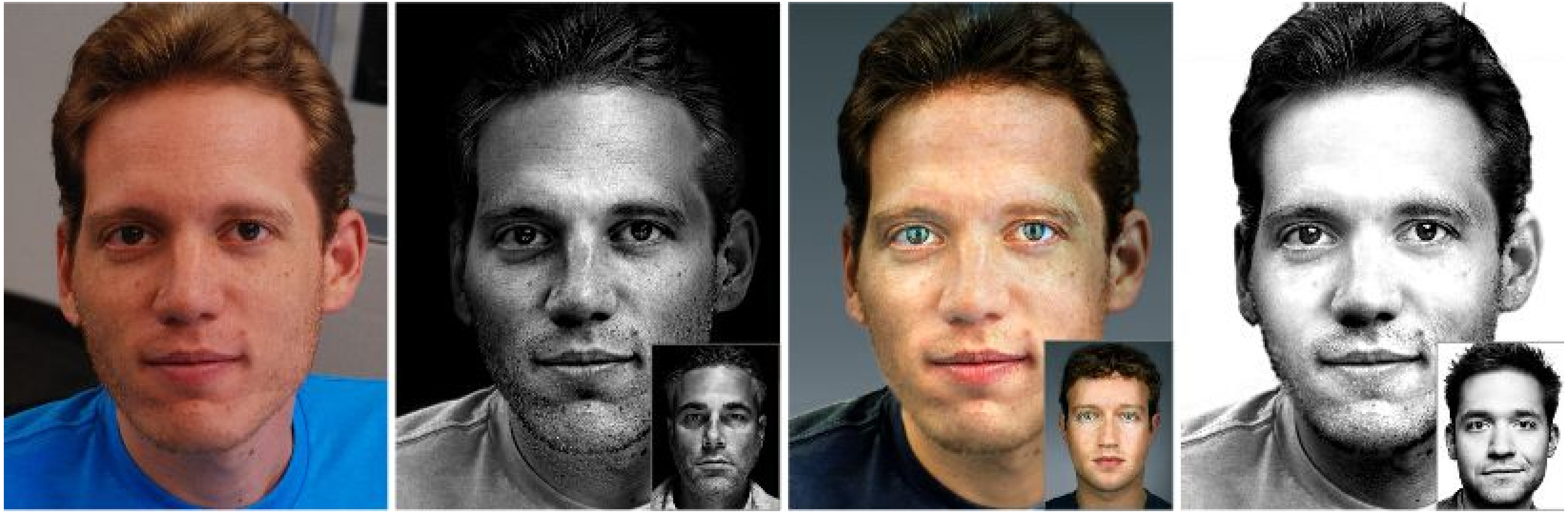
MIT CSAIL, Adobe, University of Virginia

Aim

The aim of the project is implement Style Transfer of Headshot portraits using the process of image morphing.

We first explore the process of morphing images, by implementing the warping and cross dissolve processes. We then use the above, to implement Style Headshot portraits from an example picture to the input by generating correspondence points automatically, Morph the example and combine them.

Aim



Motivation

Headshot portraits are a popular subject in photography. Large amount of time and effort is spent on editing these photos to achieve a particular style. Different styles will elicit different moods.

However, the editing process to create such renditions requires advanced skills because features such as the eyes, the eyebrows, the skin, the mouth, and the hair all require specific treatment.

Producing such renditions requires advanced editing skills beyond the abilities of most casual photographers.

Correspondence Points Generation

To obtain correspondences between the input and reference images, we take a coarse-to-fine approach, using a series of off-the-shelf tools.

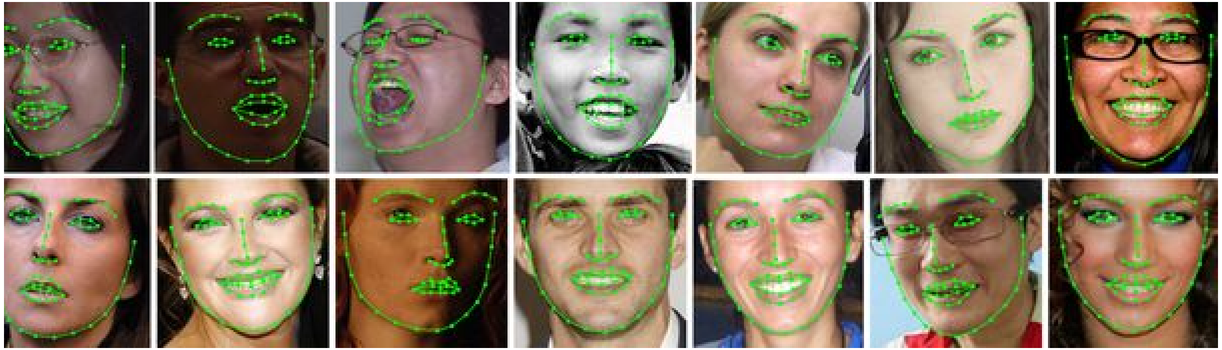
We detect the facial landmarks using a template [Discriminative Response Map Fitting (CVPR 2013) Akshay Asthana, Shiyang Cheng].

Used in the paper: Saragih et al . 2009

This gives us 66 facial landmarks as well as a triangulated face template.

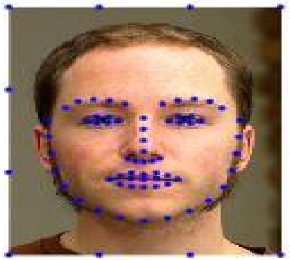
We then morph the example image to match the input image as described in the beginning.

Discriminative Response Map Fitting (DRMF 2013)

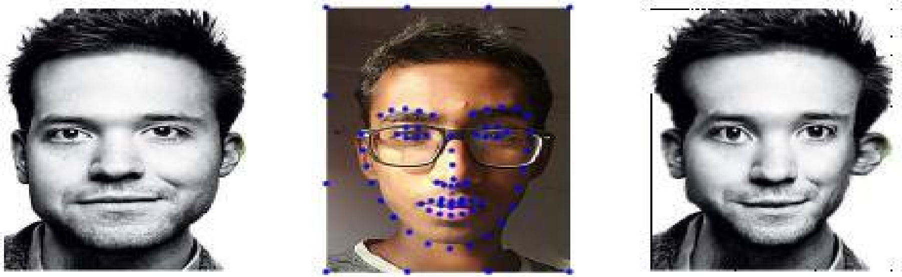


This is the Discriminative Response Map Fitting Matlab Code (CVPR 2013) written by Akshay Asthana and Shiyang Cheng. It is a fully automatic system that detects 66 landmark points on the face and estimates the rough 3D head pose. The code also contains a robust face detector that is suitable for 'wild' faces.

Correspondence Points Generation



Correspondence Points Generation



Pyramid Based Energy Transfer

We begin by constructing a Laplacian pyramid of the image, without down sampling as you proceed through the levels. After experimentation, it is found that sigma = 1.6 gives the most optimal results.

$$L_\ell[I] = \begin{cases} I - I \otimes G(2) & \text{if } \ell = 0 \\ I \otimes G(2^\ell) - I \otimes G(2^{\ell+1}) & \text{if } \ell > 0 \end{cases}$$

$$R[I] = I \otimes G(2^n)$$

The energy of an image at a certain level in the Laplacian pyramid is defined as the square of the Laplacian at that level convolved with a Gaussian filter.

$$S_\ell[I] = L_\ell^2[I] \otimes G(2^{\ell+1})$$

Pyramid Based Energy Transfer

At each level, once the energy of both the images is calculated, the gain between the energy maps of the morphed and the images are computed based on them.

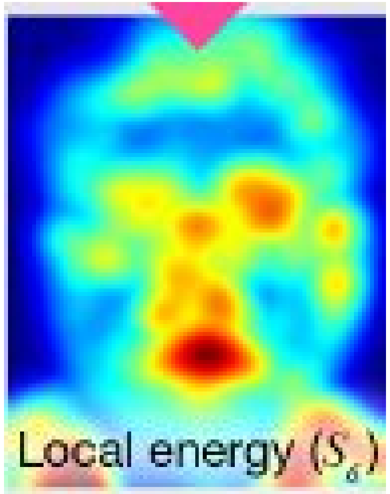
$$L_\ell[O] = L_\ell[I] \times \mathrm{Gain}$$

$$\text{with Gain} = \sqrt{\frac{\tilde{S}_\ell[E]}{S_\ell[I] + \epsilon}}$$

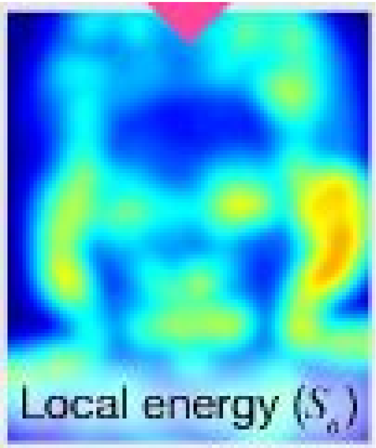
In order to reconstruct the final output image from the Laplacian Pyramid, we simply add all the elements of the pyramid, including the residual, so that the intermediate terms are cancelled and we are left with only the image.

We use the *CIE-Lab* color space.

Pyramid Based Energy Transfer



Pyramid Based Energy Transfer



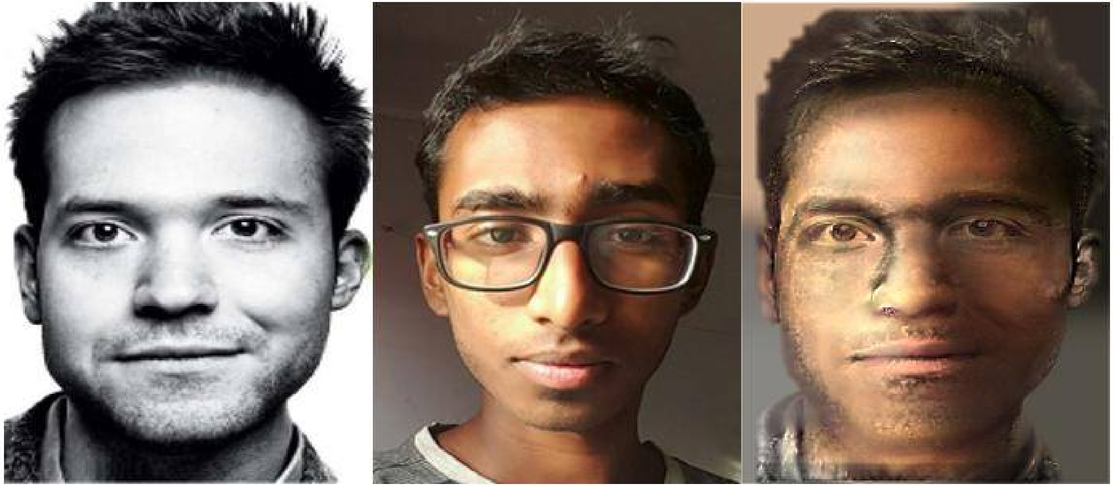
Results

Results

Results

Results

Results



Thank You!