FaceChanger for Android

Megan Van Welie
Tufts University, Computer Science Department
December 6, 2016

An open source, face swapping and morphing program for Android, which is explored as an application for face tracking, orientation alignment, and image blending.

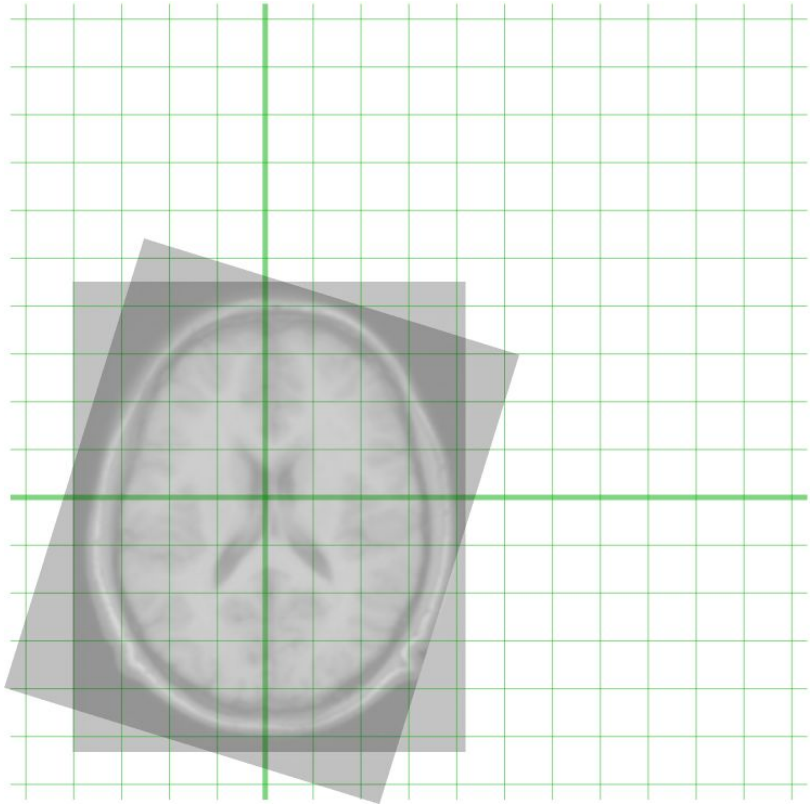
Motivation

Motivation

General techniques are applicable to a wide range of fields, including medical image processing and accessibility software.

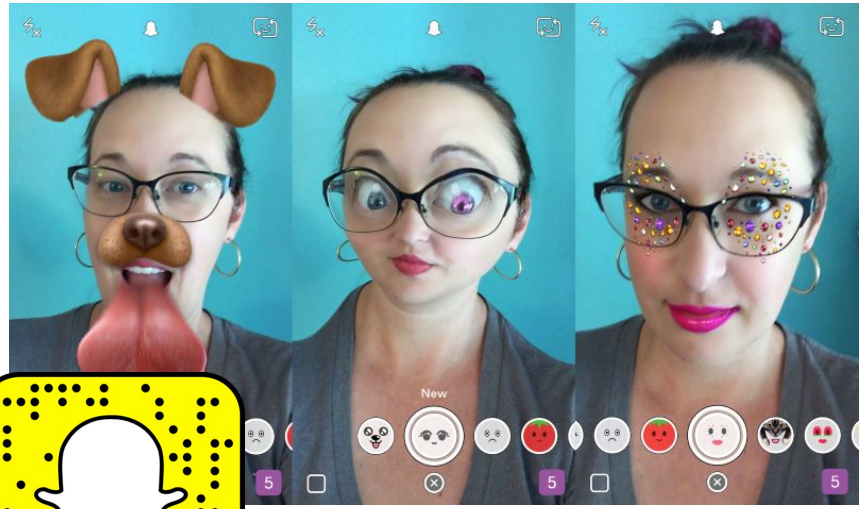
CameraMouse

Innovative software for people with disabilities



Motivation

Entertainment and Marketing



Background

Background Outline

Primary Techniques:

- Facial landmark detection
- Absolute orientation for image alignment
- Poisson image blending

Background Outline

Primary Techniques:

- Facial landmark detection
- Absolute orientation for image alignment
- Poisson image blending

Tools:

- OpenCV
- Dlib, a C++ machine learning toolkit
- Android NDK

Background

Facial Landmark Detection

- Addressed using machine learning techniques
- Dlib implementation based on:

Kazemi, Vahid, and Josephine Sullivan. "One millisecond face alignment with an ensemble of regression trees."

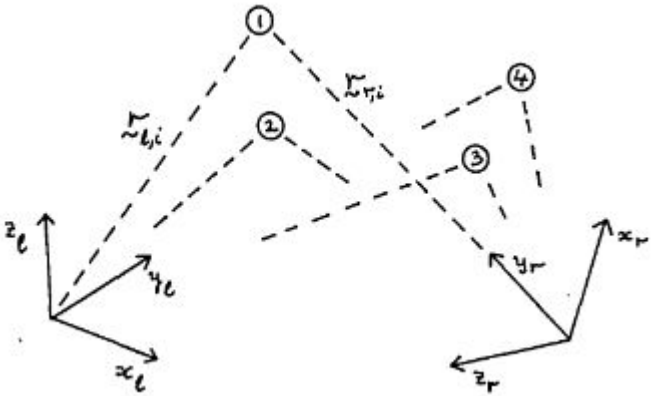
Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014.

- Identifies 68 landmarks around the face

Background

Absolute Orientation

Horn, Berthold KP. "Closed-form solution of absolute orientation using unit quaternions." *JOSA A* 4.4 (1987): 629-642.



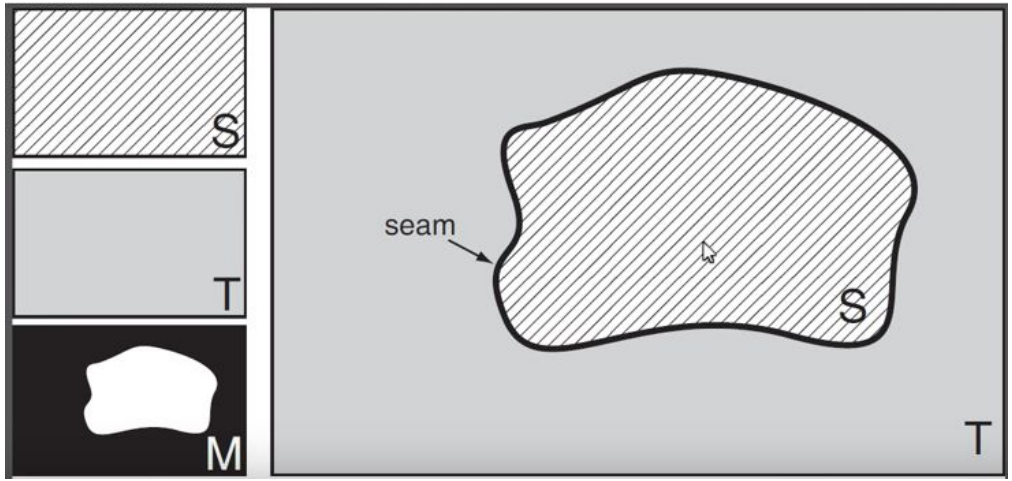
$$min_{x_0,y_0,\theta}\Sigma_{i=1}^{N}|| \vec{r_{r,i}} -(R\,\vec{r_{l,i}} - \vec{r_0})||^2$$

Solve for angle of rotation, translation, and scale difference between two sets of landmark points.

Background

Image blending

Source image: `S(x,y)`
Mask image: `M(x,y)`
Target image: `T(x,y)`
Composite image: `I(x,y)`



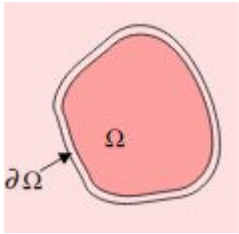
- Ideas:
  - Hard copy the source image S(x,y) onto T(x,y) where M(x,y) = 1

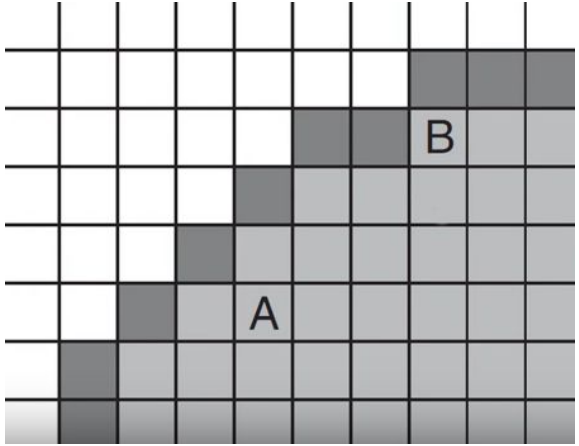
```
I(x,y) = S(x,y)M(x,y) + T(x,y)(1 - M(x,y))
```

  - Blurred transition between S(x,y) and T(x,y) along edge of M(x,y)

`Can be achieved with Gaussian and Laplacian image pyramids`

Background





Poisson Image blending

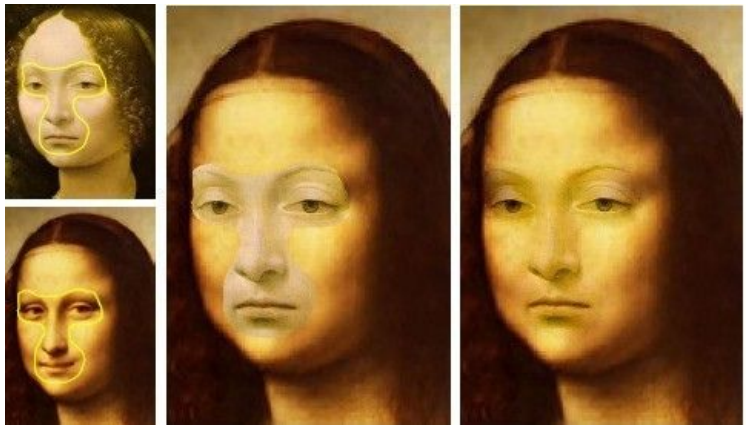
Pérez, Patrick, Michel Gangnet, and Andrew Blake. "Poisson image editing." ACM Transactions on Graphics (TOG). Vol. 22. No. 3. ACM, 2003.

$$min_{I(x,y)\in\Omega} \iint ||\nabla I(x,y) - \nabla S(x,y)||^2 dxdy$$

constrained such that

$$I(x,y) = T(x,y) \text{ on } \partial\Omega$$

- OpenCV implementation: *seamlessClone*
- Great lecture by Professor Rich Radke (RPI)

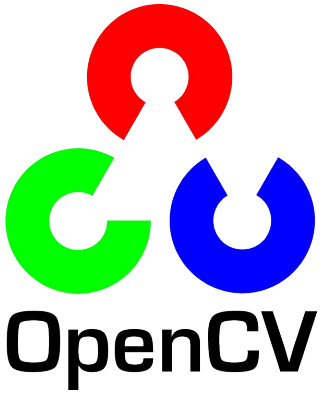


Background

Tools and how I used them

OpenCV:

- Image manipulation
- Poisson image blending



Dlib:

- Facial landmark detection

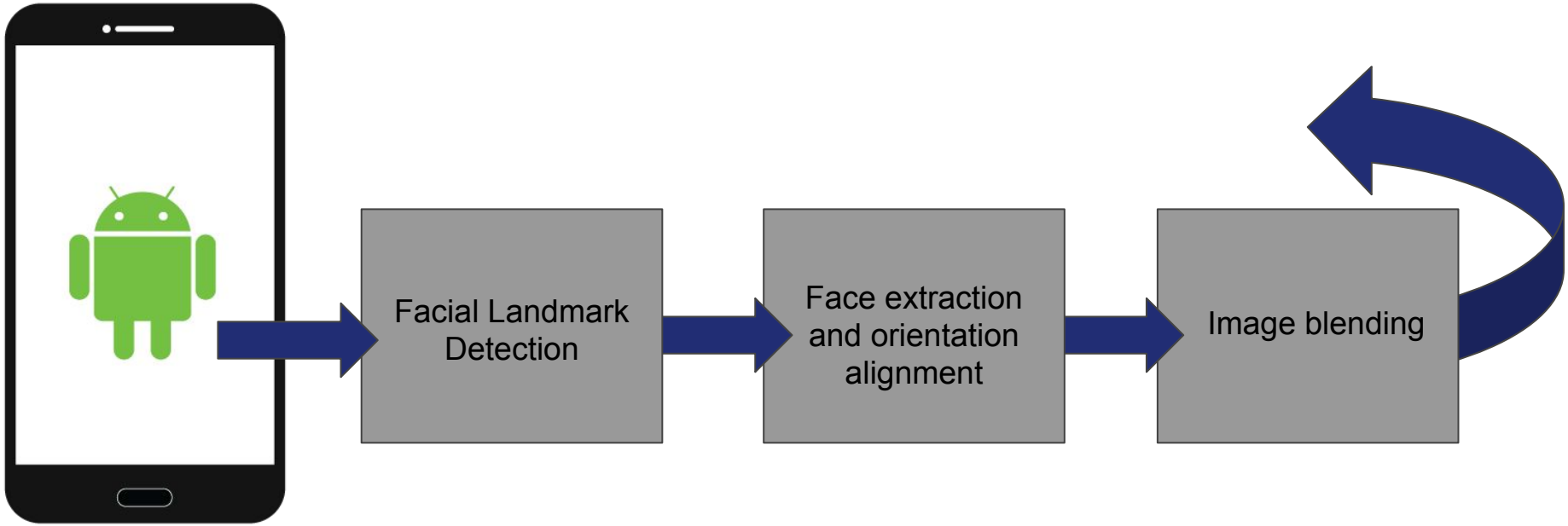
Android SDK and NDK:

- Android Native Development Kit, toolset that allows processing within application in C++

Methodology

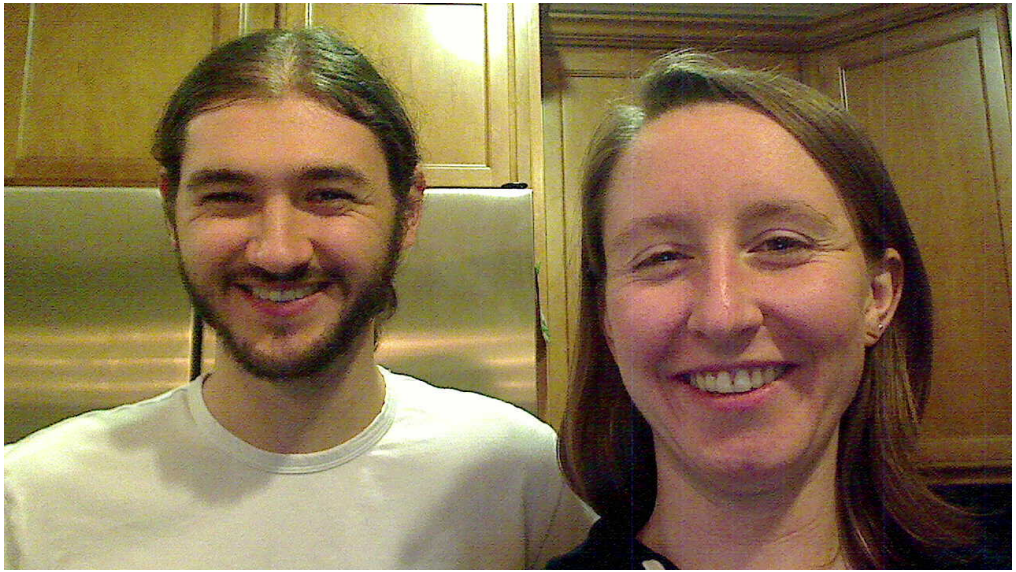
Methodology

- Developed each technique in a sandbox with ideal data
- Joined each step into a complete system



Experiments and Results

Results



Original image

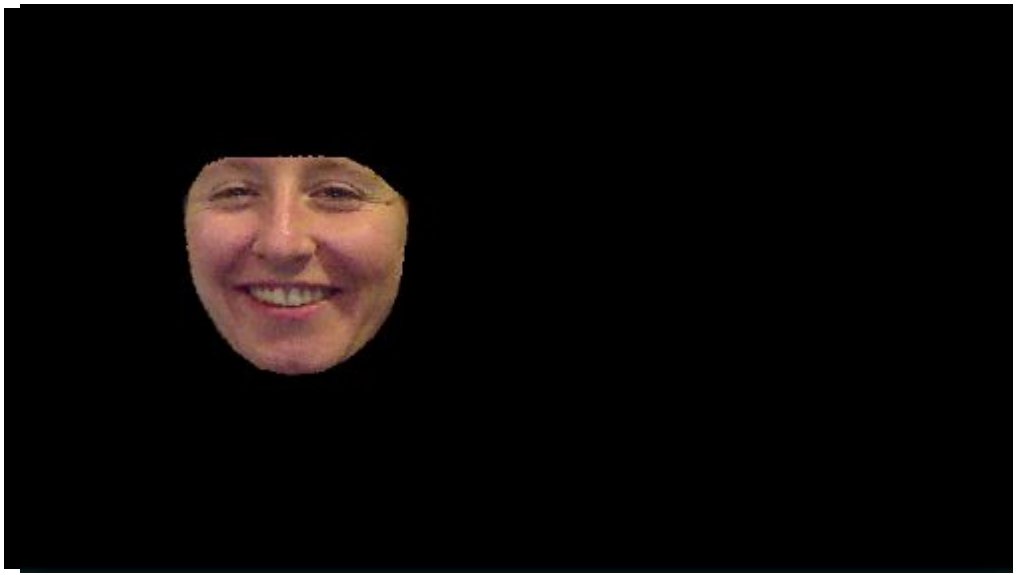
Results

Facial landmark points detected

Results

Extracted face in original position

Results



Aligned with swapped position

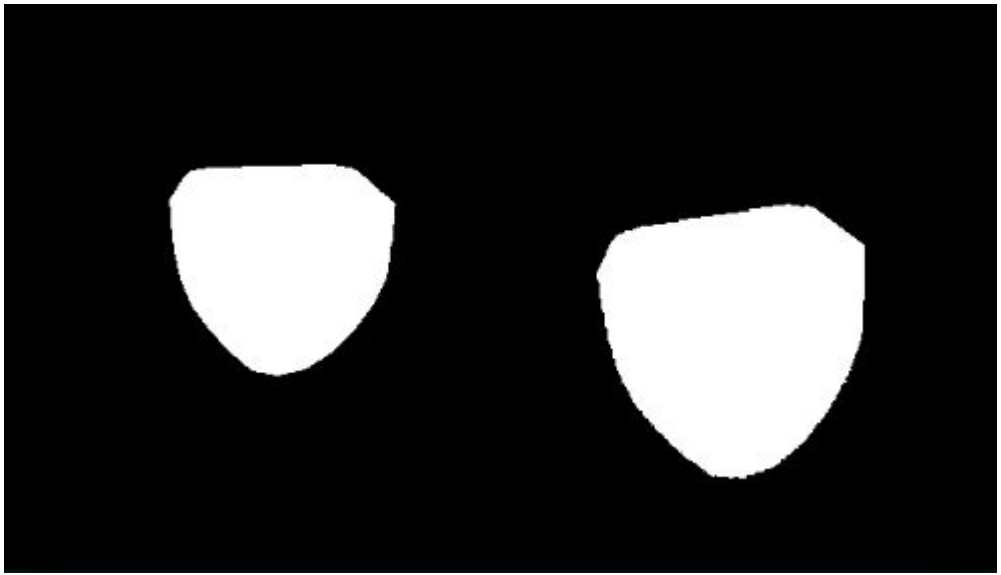
Results

Extracted face in original position

Results

Aligned with swapped position

Results



Final mask to be used with hard copy

Results

Hard copy from mask

Results

Poisson Image Blending

Conclusions

- Still working to fully port to Android (close!)
- Still working to make it run in real-time

- Improvements:
  - Blending could add additional constraints to prevent edge smudging seen when the target image has a strong edge (as in the case of the beard).
  - Real-time processing is not quite achieved.
  - Color blending between images for a more realistic result.

- Takeaways:
  - Even simple tasks still take a lot of forethought to achieve desired results.
  - Computational photography looks like an interesting field that I would love to explore.