Team APPY FIZZ

1. Ahish Deshpande (AD) (2018102022, ECE)
2. Pranav Kirsur (PK) (2018101070, CSE)
3. Pranav Tadimeti (PT) (2018101055, CSE)
4. Yoogottam Khandelwal (YK) (2018101019, CSE)

Friend Blend

Mentor TA: Adhithya Arun

Repo URL: https://github.com/Digital-Image-Processing-IIITH/project-appy-fizz

Problem Description

Problem Description

1. Two friends don’t want to take a selfie, maybe because of camera resolution.
2. Want to have a picture together, nobody around to take it.
3. Two people visit the same place at different times but want to have a picture together at that place.

Project Paper Introduction

Project Paper Introduction

(g) Image 1 (h) Image 2 (i) Result

(j) Image 1 (k) Image 2 (l) Result

(m) Image 1 (n) Image 2 (o) Result

Given two input images, our goal is to create a third image with both Person A and Person B in the photo together.

The FriendBlend algorithm mainly relies on segmentation and registration techniques for extraction and merging images.

Project Pipeline

Process

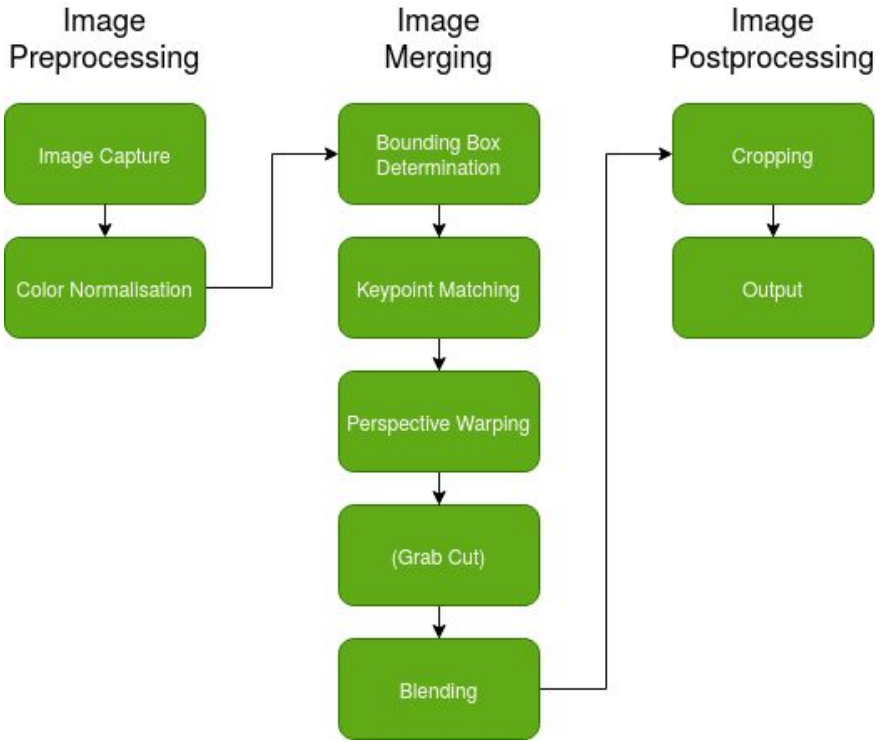


Image Preprocessing
Image Merging
Image Postprocessing
Image Capture
Color Normalisation
Bounding Box Determination
Keypoint Matching
Perspective Warping
(Grab Cut)
Blending
Cropping
Output

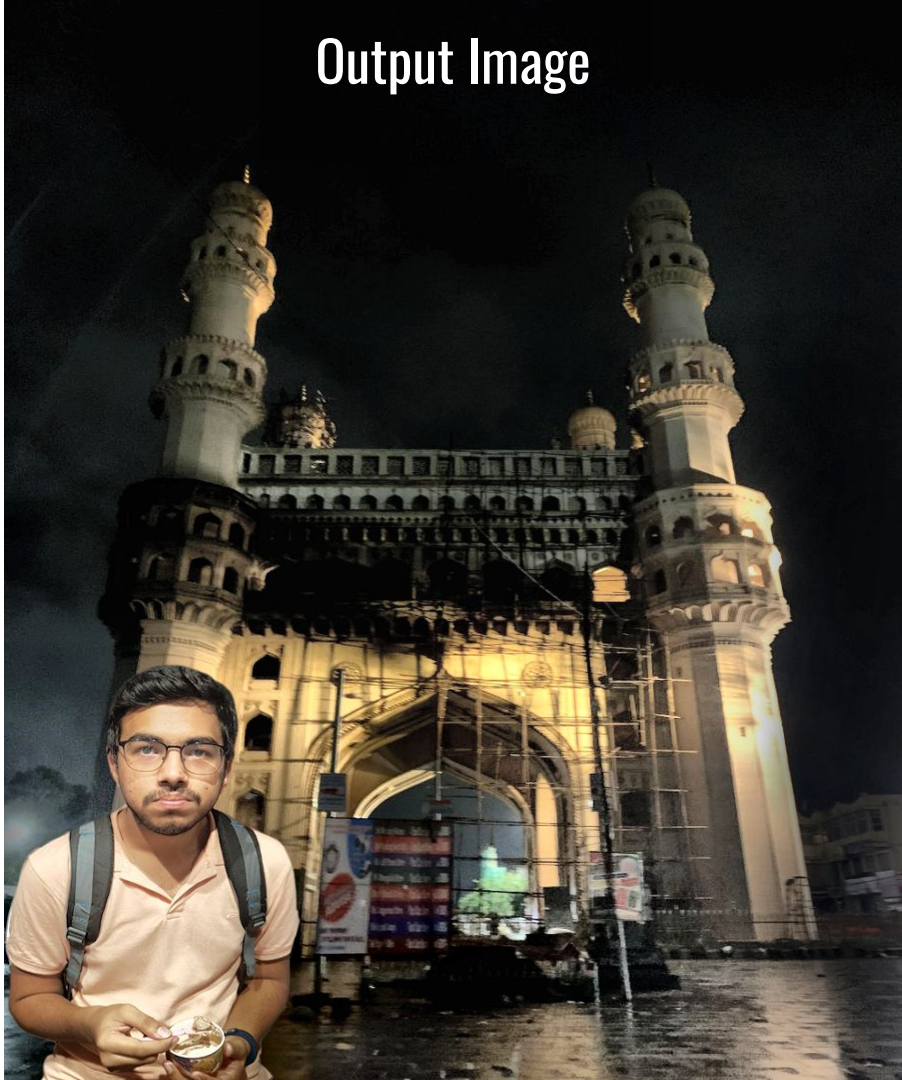
Stage 1: Color Correction

Contrast Limited Adaptive Histogram Equalization

1. It is crucial to make sure that the lighting from the two input images are approximately the same before the blending process to make the final image look realistic
2. This is especially important as the colors in the image affect the homography output

Input Image

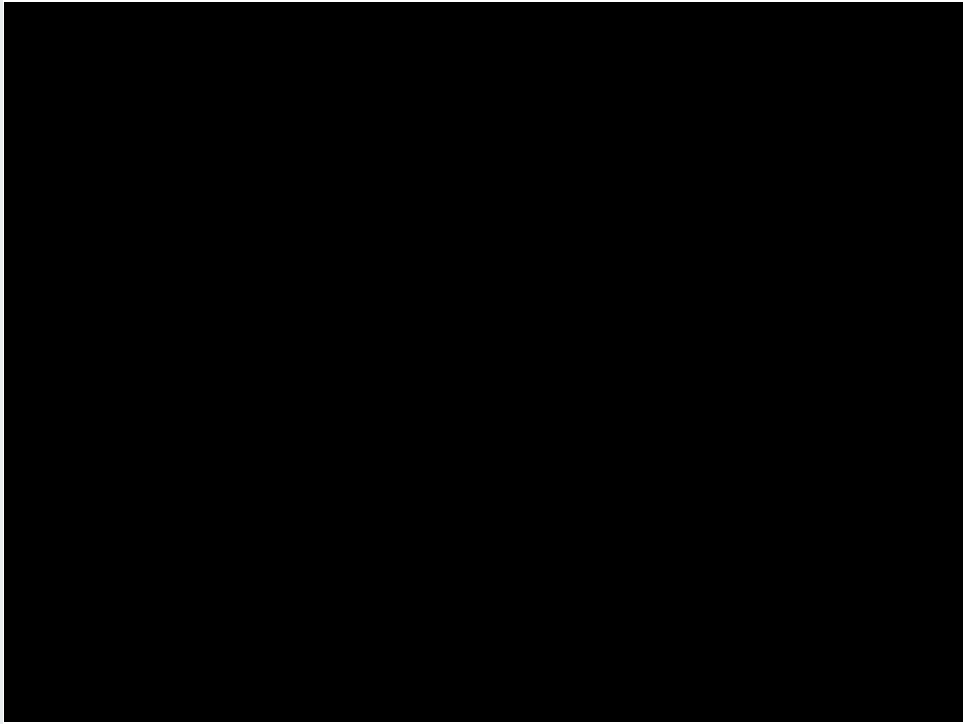
Output Image



Stage 2: Face and Body Detection

Face Detection

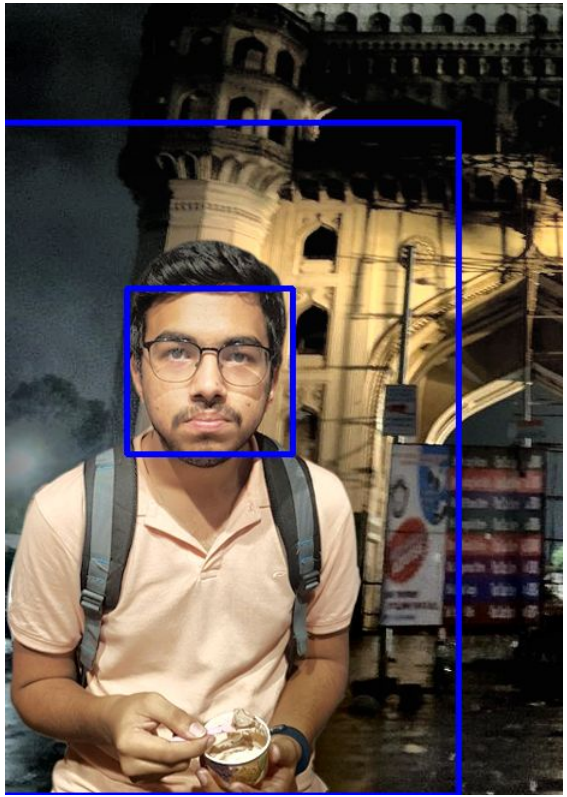
with Haar Cascade



Body Bounding Box

inferred from face

$$
\begin{aligned}
x_{left}^{body} &= x_{left}^{face} - w \\
x_{right}^{body} &= x_{left}^{face} + 2 \cdot w \\
y_{top}^{body} &= y_{top}^{face} - h \\
y_{bottom}^{body} &= height(image)
\end{aligned}
$$



Stage 3: Homography Estimation

Keypoint Detection and Matching

We use ORB features

1. Keypoint filtering based on bounding boxes and distance in the image
2. Keypoint matching based on Hamming distance

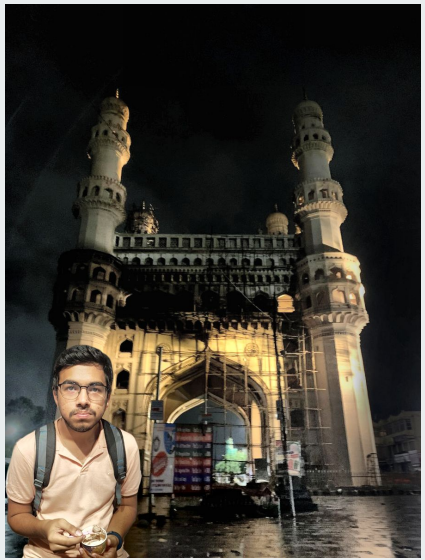
Stage 4: Image Blending

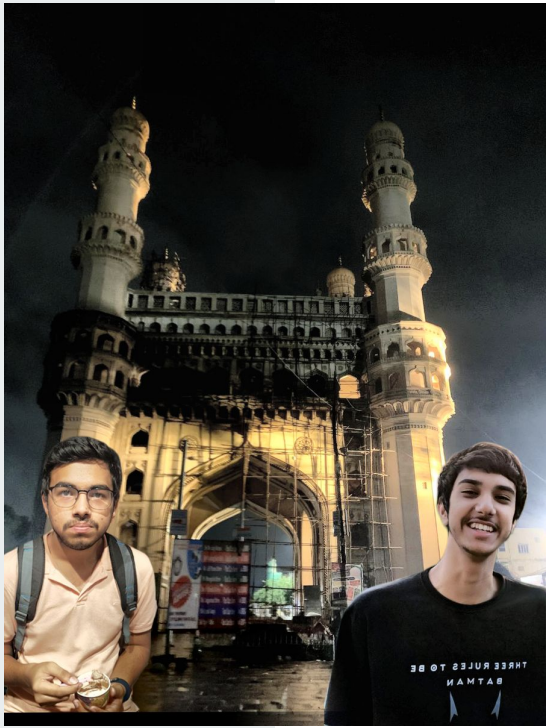
Alpha Blending

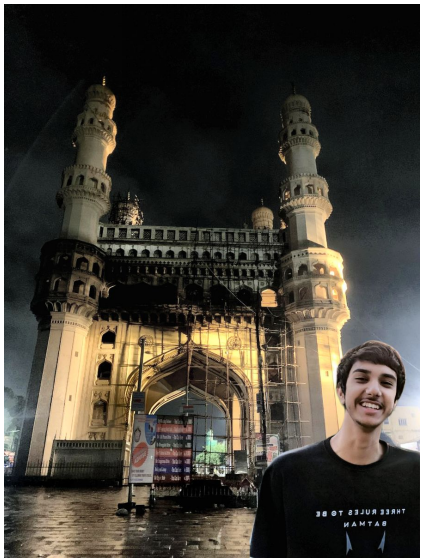
Adopted when subjects are far apart

$$\text{stepSize} = \frac{1}{\text{col End} - \text{col Start}}$$

$$i^{\text{result}}(x,y) = (1 - \text{stepCount} * \text{stepSize}) * i^{\text{left}}(x,y) + (\text{stepCount} * \text{step Size}) * i^{\text{right}}(x,y)$$





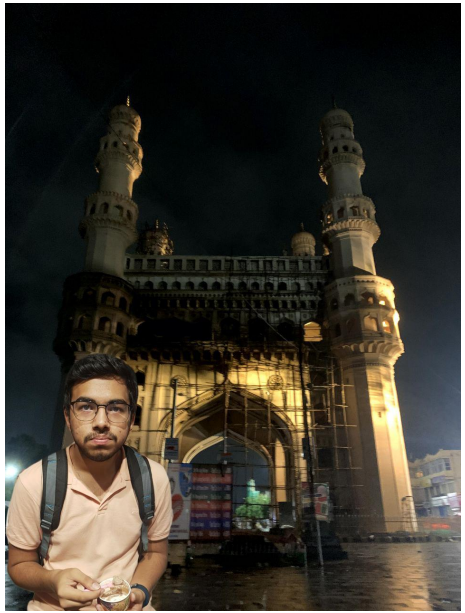


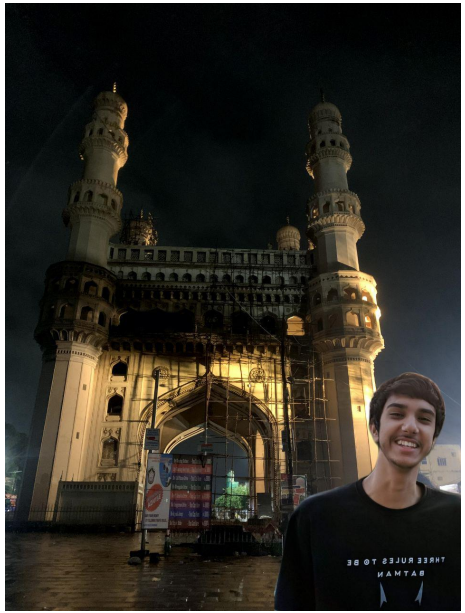
GrabCut

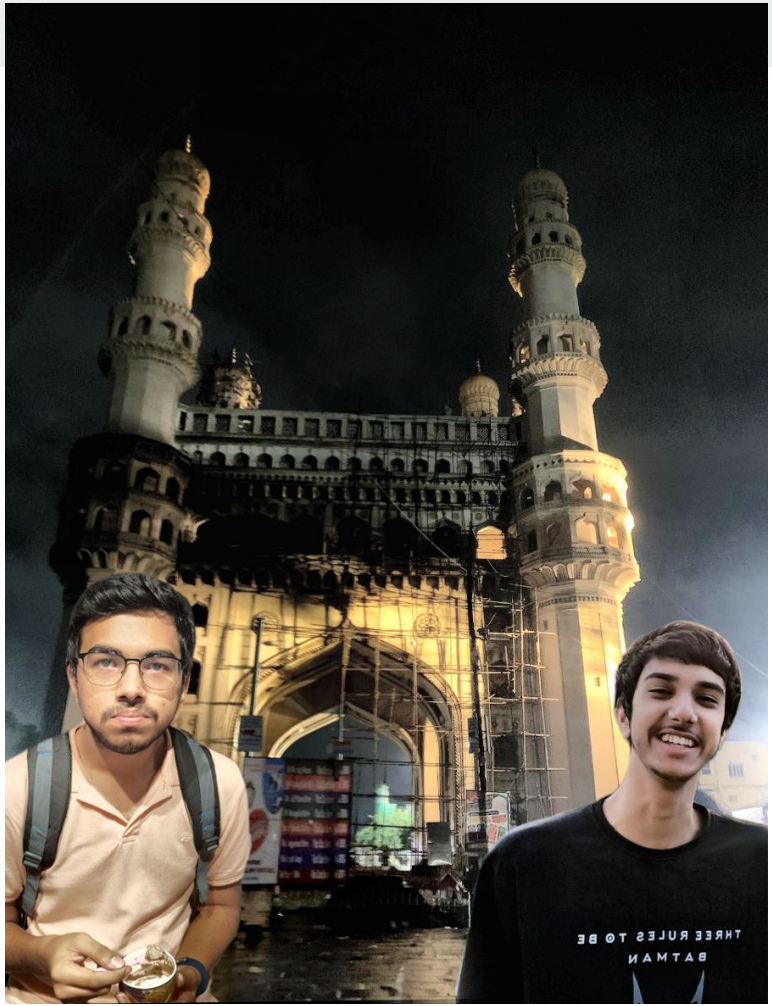
Used when subjects are very close to each other

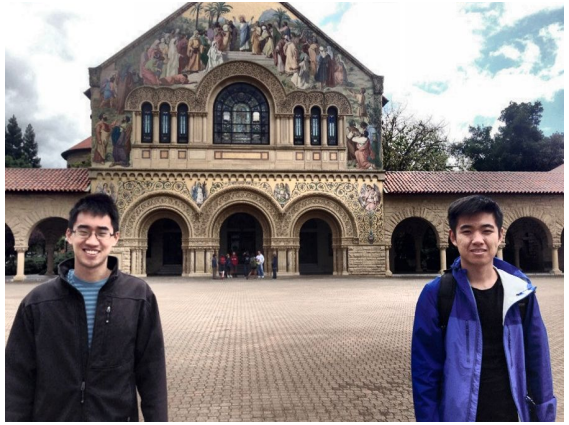
1. Initial mask for GrabCut constructed from bounding boxes.
2. Face labelled as definite foreground.
3. Area below face labelled as probable foreground.

Results

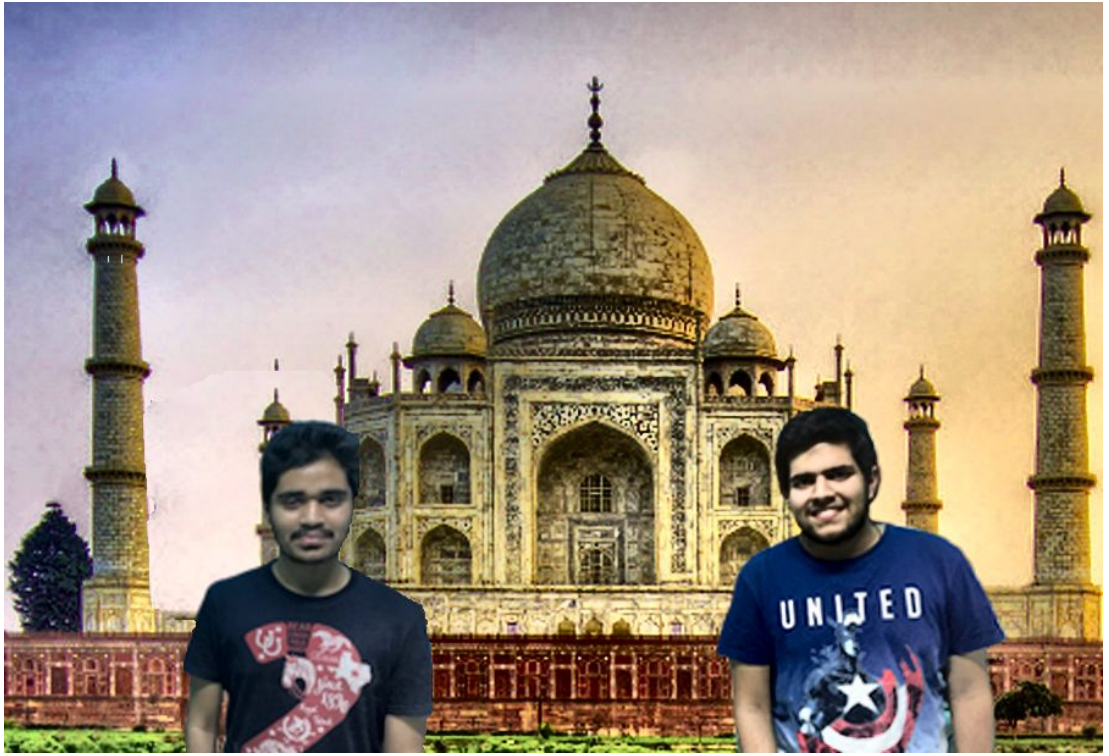



THREE RULES TO BE
BATMAN


THREE RULES TO BE
BATMAN




UNITED


UNITED


BEAR CREEK POLO RANCH
Since 1890


BEAR CREEK POLO RANCH
Since 1890

Limitations

It's not perfect!

1. Face detection is not always perfect. Possible solutions:
   a. Swap haarcascade with current SOTA, RetinaFace: Single-stage Dense Face Localisation in the Wild [code]
2. Homography is not always computable or incorrect. Possible solutions:
   a. Using optical flow for matching the perspective in both images
3. GrabCut doesn't work perfectly. Possible solutions:
   a. Swap GrabCut with current SOTA, [Hierarchical Multi-Scale Attention for Semantic Segmentation] [code]
   b. Calculate more accurate mask by having minimal user input
4. Sometimes Color Correction gives a makeup-ish feel to the face
5. Only works with 2 images, with a single face in each of them

Examples

Face detection failing

Incorrect Homography

GrabCut leaves artifacts

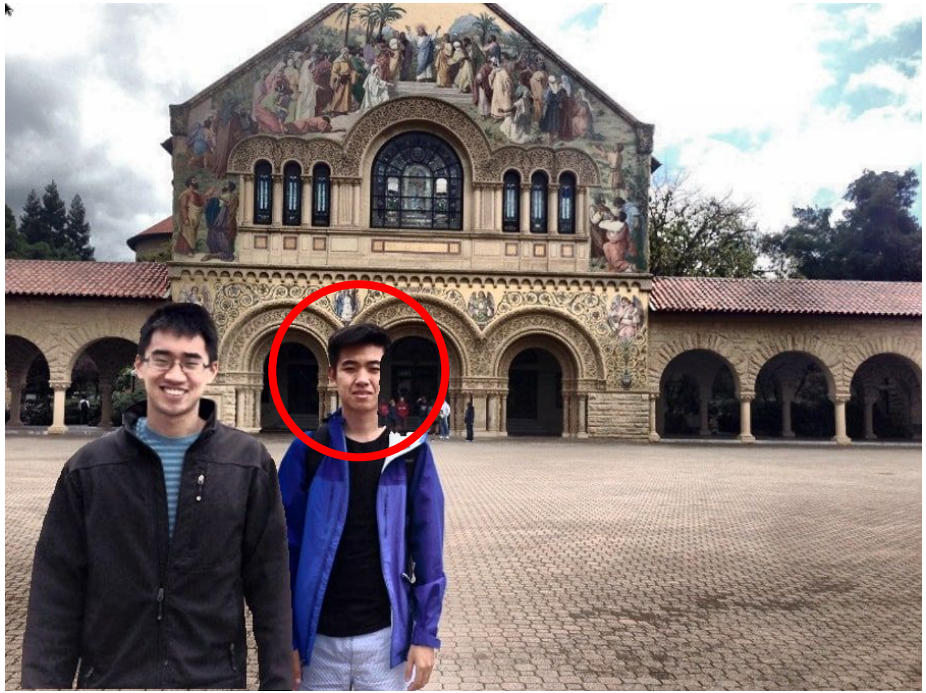
Color Correction in presence of extra light

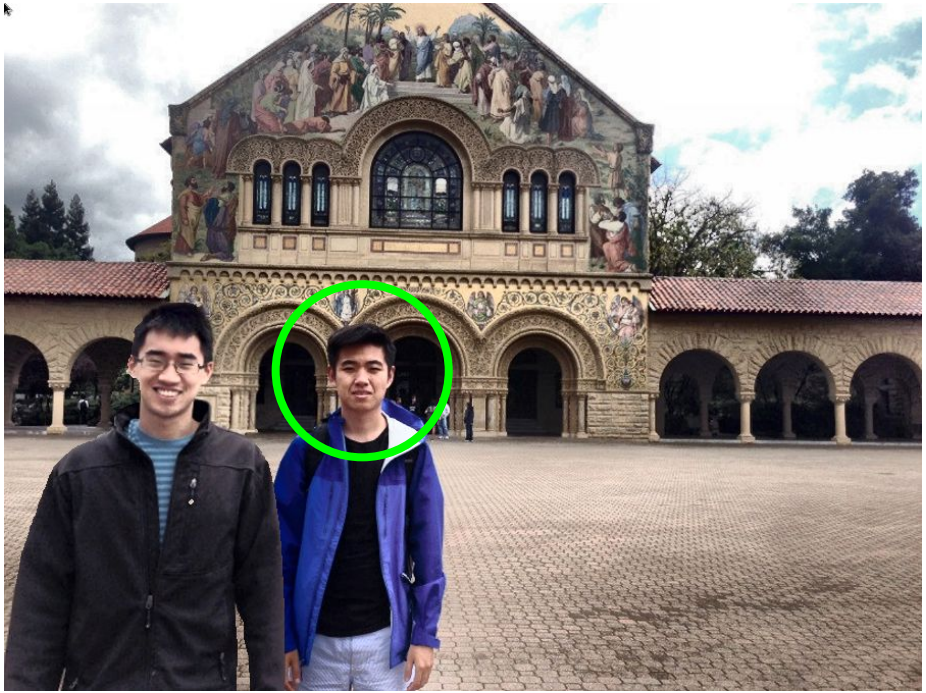
Conclusion

It works!

1. In most of the cases the results are very realistic.
2. We mostly stuck to the paper's details and parameters but we deviated from it in a few cases to get better results.
    a. Added a small margin to face bounding box mask in GrabCut, to ensure hair is not cut off.
    b. Connected components on mask with face center as seed
3. We achieved 2x speedup by parallelizing the independent steps in the pipeline.
4. Although we implemented GrabCut, it took about 60x more time (4s -> 4m) so we ended up using library cv.grabCut instead

GrabCut result improvement





Contribution

1. Ahish Deshpande (AD)
2. Pranav Kirsur (PK)
3. Pranav Tadimeti (PT)
4. Yoogottam Khandelwal (YK)

1. Color correction: **AD**
2. Face and Body detection: **PT**
3. Keypoint matching and Homography: **YK, PK**
4. Blending:
   a. Alpha Blending: **AD**
   b. GrabCut: **AD, PK, PT**
   c. GrabCut result improvement: **YK**
5. Cropping: **PT**
6. Runtime Optimization: **YK**
7. Limitations and Analysis: **ALL**
8. Project Presentation: **ALL**