MATLAB ChromaKey, *in VIDEO*

Project Team: Nkiruka Chuka-Obah, nkiruka0@stanford.edu

Introduction:

Chroma Key is a popular video/photo editing technique that allows for intermixing different video/static images. It involves the superposition of a foreground image on a desired background. It has many applications in television broadcasting, as well as many photo editing applications. However, Chroma Key applied to video editing has several drawbacks. For instance, most video images have to be filmed with a solid color (usually blue or green) background as this allows for easy detection of human usually posed before the image. In addition, intensive user interaction is required to replace the background image with the desired background image to create the desired effect.

Description and Goals:

This project is a MATLAB implementation of Chroma Key video editing that attempts to solve the background segmentation problem. It will employ automatic detection and/or tracking of the person(s) in the foreground image of a video sequence, hence facilitating automatic replacement of the background with the user provided background image in the video frames. In addition, it will remove the necessity for a solid color background during the video recording process, since solid color backgrounds are only necessary to properly segment out the human speaking in the video.

The phases (or goals) of this project are as follows, with each step culminating in the final project:

a) Develop a simple MATLAB GUI to implement static Chroma Key with user-driven input. The user provides a picture to be modified (modInput), and a sample of the background image (backInput). The image, modInput, will have a solid color background. The GUI produces a modified picture with the background of modInput changed to that provided in backInput. In this stage, the goals will be to develop the following algorithms:
    a. Automatic segmentation of the person(s) in the image, modInput, using skin color, human pose, and/or face detection as parameters for the segmentation.
    b. User-driven segmentation of the person(s) in the image, modInput, with the use of markers to enhance the automatic results and provide a better user desired output. The user may place markers around the person(s) wanted in the final image.

b) Enhance the GUI to take video inputs as well as static images for foreground/background image.

c) Enhance the GUI by applying image segmentation techniques to detect faces, or human poses, and track them in a set of video frames. Possible techniques are discussed in the provided references. A particular technique has not been decided on as yet. Include user driven inputs via the use of markers to enhance the segmentation process.

Work Accomplished:

A MATLAB program has been implemented that performs simple Chroma Key editing. It takes as input a picture with a solid color background, and a picture with the desired background. It produces as output an image with the foreground of the solid color image, and the desired background. It is shown below with the results, and code provided in the appendices.

Figure 1: Foreground Image, Background Image

Figure 2: Combination Image

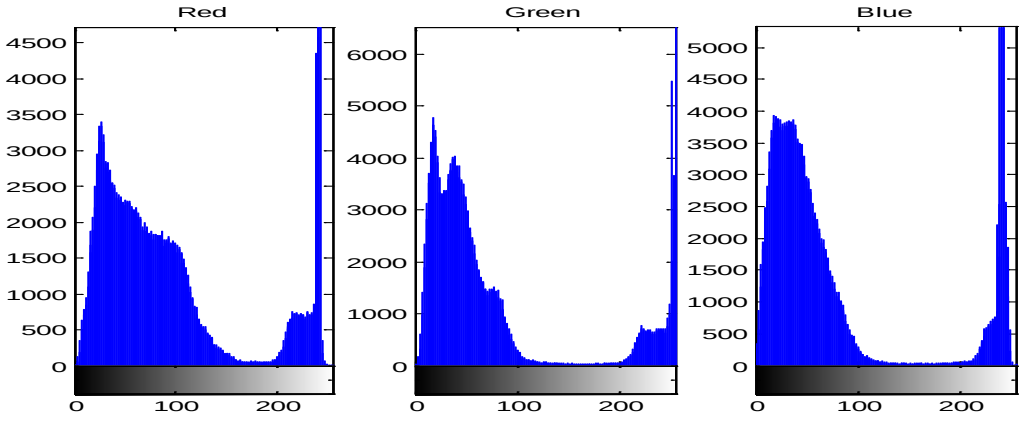

Figure 3: Histograms of the Red, Green and Blue components of the foreground image.

The result, shown in **Figure 2**, is achieved by applying a threshold to the foreground image (**Figure 1)** to remove the mostly white background of the foreground image, creating a foreground mask. The inverse of this foreground mask is multiplied by the foreground image. This result is added to the product of the foreground mask with the background image (**Figure 1)**, creating the result of **Figure 2.** The threshold was obtained by analyzing the histograms of the Red, Green and Blue components of the foreground image to detect the pixels of the mostly white background of the foreground. A threshold of ~200 is used. No automatic algorithms are applied.

References:

Ahmed, R.; Karmakar, G.C.; Dooley, L.S.; , "Automatic video background replacement using shape-based probabilistic spatio-temporal object segmentation," *Information, Communications & Signal Processing, 2007 6th International Conference on* , vol., no., pp.1-4, 10-13 Dec. 2007

Xiao-Yan Zhang; Rong-Chun Zhao; , "Automatic Video Object Segmentation Using Wavelet Transform and Moving Edge Detection," *Machine Learning and Cybernetics, 2006 International Conference on* , vol., no., pp.1174-1177, 13-16 Aug. 2006

Raja, Y.; McKenna, S.J.; Shaogang Gong; , "Tracking and segmenting people in varying lighting conditions using colour," *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on* , vol., no., pp.228-233, 14-16 Apr 1998

Chen, D.; Denman, S.; Fookes, C.; Sridharan, S.; , "Accurate Silhouettes for Surveillance - Improved Motion Segmentation Using Graph Cuts," *Digital Image Computing: Techniques and Applications (DICTA), 2010 International Conference on* , vol., no., pp.369-374, 1-3 Dec. 2010

Appendix:

```
%function imageReturn = chromaKey(foregroundFile,backgroundFile)
key = [200,200,200]; %Color key for R,G,B -- Automatically detect later
foregroundFile = 'fatherSon.jpg';%white background
backgroundFile = 'vineyard.jpg';
%Color Images
foreground = imread(foregroundFile);
background = imread(backgroundFile);
background = imresize(background, [size(foreground,1) size(foreground,2)]);

%Show Input Images
figure('Name','Foreground, and Background Images')
subplot(1,2,1)
imshow(foreground,[0 255]);
title('Foreground Image');
subplot(1,2,2)
imshow(background,[0 255]);
title('Background Image');

%Separate out R, G, B
Rforeground = foreground(:,:,1);
Gforeground = foreground(:,:,2);
Bforeground = foreground(:,:,3);

%Get histograms for each componet.
Rhistogram = imhist(Rforeground);
Ghistogram = imhist(Gforeground);
Bhistogram = imhist(Bforeground);

%Plot histogram
figure('Name','Histogram of Foreground Image')
subplot(1,3,1)
```

```
imhist(Rforeground);
title('Red')
subplot(1,3,2)
imhist(Gforeground);
title('Green')
subplot(1,3,3)
imhist(Bforeground);
title('Blue')

%Get masks for R,G,B of foreground
%Values picked by analyzing foreground image
RforegroundMask = ones(size(Rforeground));
RforegroundMask(Rforeground<key(1)) = 0;
GforegroundMask = ones(size(Gforeground));
GforegroundMask(Gforeground<key(2)) = 0;
BforegroundMask = ones(size(Bforeground));
BforegroundMask(Bforeground<key(3)) = 0;


%final Image mask
finalImagemask =
cast(RforegroundMask.*GforegroundMask.*BforegroundMask,'uint8');
finalImagemaskinv = cast(not(finalImagemask),'uint8');

%final Image
finalImage = zeros(size(background));
finalImage(:,:,1)= finalImagemask.*background(:,:,1)+
finalImagemaskinv.*foreground(:,:,1);
finalImage(:,:,2)= finalImagemask.*background(:,:,2)+
finalImagemaskinv.*foreground(:,:,2);
finalImage(:,:,3)= finalImagemask.*background(:,:,3)+
finalImagemaskinv.*foreground(:,:,3);

figure('Name','Final Image')
imshow(cast(finalImage,'uint8'));
```