Colorado School of Mines

Computer Vision

Professor William Hoff
Dept of Electrical Engineering & Computer Science
http://inside.mines.edu/~whoff/
Homography Applications
Example – Creating a Panorama

• The images below were taken with a camera that has rotated (but not translated)

• A projective transform (homography) can align the images so they can be merged into a panorama, as if they were all taken by the camera in the center position

• Strategy: find a projective transform to register the left image to the center image, and also a projective transform to register the right image to the center image.
Useful Matlab functions

• **cpselect**
  – Brings up a GUI to allow user to select corresponding control, or tie points
  – Example:
    • \[[\text{Pts}_1, \text{Pts}_2] = \text{cpselect}(I_1, I_2, 'Wait', \text{true});\]

• **fitgeotrans**
  – Given two sets of control points, estimate the image transform using least squares fitting
  – Example:
    • \[t_{12} = \text{fitgeotrans}(..., 'projective');\]

• **imwarp**
  – Transform an image using a pre-defined transform
  – Example:
    • \[\text{I}_{\text{Trans}} = \text{imwarp}(I_1, t_{12}, 'OutputView', \text{ref2Dinput});\]
Example - panorama

• Find a projective transform to transform the left image into the coordinate system of the center

```matlab
I2 = imread('pavilionCenter.jpg');
[H,W] = size(I2);

I1 = imread('pavilionLeft.jpg');
I3 = imread('pavilionRight.jpg');

% Start the GUI to select corresponding points
[Pts1,Pts2] = cpselect(I1,I2, 'Wait', true);

% Compute transform, from corresponding control points
t12 = fitgeotrans(Pts1,Pts2,'projective');
```

• Find a projective transform to transform the right image into the coordinate system of the center

```matlab
% Start the GUI to select corresponding points
[Pts3,Pts2] = cpselect(I3,I2, 'Wait', true);

t32 = fitgeotrans(Pts3,Pts2,'projective');
```
You need at least four points – more is better. Here are the control points I picked between the left and center images:

Here are the control points between the center and right images:
Saving points to a file

After finding the points, you might want to save them, so that you don’t have to find them again, every time you run the program.

clear all
close all

I2 = imread('pavilionCenter.jpg');
[H,W] = size(I2);

I1 = imread('pavilionLeft.jpg');
I3 = imread('pavilionRight.jpg');

if ~exist('points12_pavilion.mat', 'file')
    % Start the GUI to select corresponding points
    [Pts1,Pts2] = cpselect(I1,I2, 'Wait', true);
    save('points12_pavilion.mat', 'Pts1', 'Pts2');
else
    load('points12_pavilion.mat');
end

% Compute transform, from corresponding control points
t12 = fitgeotrans(Pts1,Pts2,'projective');

if ~exist('points32_pavilion.mat', 'file')
    % Start the GUI to select corresponding points
    [Pts3,Pts2] = cpselect(I3,I2, 'Wait', true);
    save('points32_pavilion.mat', 'Pts3', 'Pts2');
else
    load('points32_pavilion.mat');
end

% Compute transform, from corresponding control points
t32 = fitgeotrans(Pts3,Pts2,'projective');
Example - panorama

ref2Dinput = imref2d( ... 
    [H, 3*W], ... % Size of output image (rows, cols)
    [-W, 2*W], ... % xWorldLimits
    [1, H]); % yWorldLimits

I1Warp = imwarp(I1,t12, 'OutputView', ref2Dinput );
I3Warp = imwarp(I3,t32, 'OutputView', ref2Dinput );

Icombined = [I1Warp(:,1:W) I2 I3Warp(:,2*W+1:3*W)];
figure, imshow(Icombined, []);

As the red line indicates, the lines of the building are straight now, across the combined image. Note – although the images are now geometrically matched, you can see a difference in lighting between the images as they are stitched together. There are methods for blending images together, such as using a multiresolution spline.
Example – Replace a photo with another

• Professor Shoureshi is no longer at CSM. Let’s say that we want to replace his photo with a photo of Dr. Steele.
• We can find the homography (projective transform) that will map Dr. Steele’s photo to the correct position in the first image.
Example – replace photo

• First find the four pairs of corresponding points between the two photos
  – We’ll use Matlab’s selector tool, cpselect
  – After finding the points, you might want to save them, so that you don’t have to find them again, every time you run the program

```matlab
clear all, close all
I1 = imread('wall1.jpg');
I2 = imread('wall2.jpg');

% Find 4 corners of picture.
if ~exist('points.mat', 'file')
    % Start the GUI to select corresponding points
    [Pts1,Pts2] = cpselect(I1,I2, 'Wait', true);
    save('points.mat', 'Pts1', 'Pts2');
else
    load('points.mat');
end
```
Example – replace photo

• Show the picked points to verify that we got them correctly.

% Display for verification
imshow(I1, []);
for i=1:size(Pts1,1)
    rectangle('Position', [Pts1(i,1)-4 Pts1(i,2)-4 8 8], 'EdgeColor', 'r');
    text(Pts1(i,1), Pts1(i,2), sprintf('%d', i));
end
figure, imshow(I2, []);
for i=1:size(Pts2,1)
    rectangle('Position', [Pts2(i,1)-4 Pts2(i,2)-4 8 8], 'EdgeColor', 'r');
    text(Pts2(i,1), Pts2(i,2), sprintf('%d', i));
end
Example – replace photo

• Compute the projective transform, and then use it to warp image 2 to match image 1
• That should put the new photo in the correct position to replace the old photo

% Transform image 2 to image 1. Now the new picture is in
% the right place to replace the old picture.
T21 = fitgeotrans(Pts2, Pts1, 'projective');
I2warp = imwarp(I2, T21, 'OutputView', ... 
    imref2d(size(I1), [1 size(I1,2)], [1 size(I1,1)]), ... 
    'Interp', 'cubic');
figure, imshow(I2warp, []);
Example – replace photo

• We want to replace only those pixels that are in the photo.
• Let’s create a mask for the area of the photo, such that each pixel = 1 if it is in the photo, and 0 if it is not.

```matlab
% We need to create a polygon mask for the area of the photo.
Iphoto = poly2mask(Pts1(:,1), Pts1(:,2), size(I1,1), size(I1,2));
Ibackground = ~Iphoto; % Also for the background
figure, imshow(Iphoto), title('Mask for photo');
figure, imshow(Ibackground), title('Mask for background');
```
Example – replace photo

• Combine original image with image of new photo.

```matlab
% The masks should be the same type as the input image.
Iphoto = uint8(Iphoto);
Ibackground = uint8(Ibackground);

% Combine pictures
if size(I1,3) == 1
    % Grayscale image
    Icombined = I1 .* Ibackground + I2warp .* Iphoto;
elseif size(I1,3) == 3
    % RGB image
    Icombined(:,:,1) = I1(:,:,1) .* Ibackground + I2warp(:,:,1) .* Iphoto;
    Icombined(:,:,2) = I1(:,:,2) .* Ibackground + I2warp(:,:,2) .* Iphoto;
    Icombined(:,:,3) = I1(:,:,3) .* Ibackground + I2warp(:,:,3) .* Iphoto;
else
    fprintf('Unknown color model\n');
end

figure, imshow(Icombined, []);
```