Project 2

Ziyang Wang
73086246

November 26, 2018

Instructions

- Panorama stitching using SIFT detector and RANSAC fit.
- Using vlfeat to detect SIFT keypoints and their descriptors. Matching descriptors by calculating their Euclidean distances. Computing affine matrix by solving Ax=b. Using RANSAC to give a robust estimate of affine transformation matrix. Using transformation matrices and thrir pseudoinverse matrices to stitch multiple images.

match()

```
function [matches] = match(sift1, sift2)
    n=size(sift1,2); %column number of descriptor1
    m=size(sift2,2); %column number of descriptor2
    matches=[];
        for i=1:n
            D1=repmat(sift1(:,i),1,m); %Get m replicates of ith column of sift1
            distance=sqrt(sum((D1-sift2).^2)); %calculate distance between this point and every keypoint of sift2
            distance_sort=sort(distance); %find the point with minimum distance
            if distance_sort(1)<=0.8*distance_sort(2) %If the distance is more than 0.8 between first neighbor and second neighbor, ignore this match.
                j=find(distance==distance_sort(1),1);
                matches=[matches;[i j]];
            end
        end
end
```

RANSACFit

If I write convolution implement as below:

```
n=Inf;
for i=1:maxIter
    [D1,D2]=part(M,seedSetSize); %Randomly choose S keypoints.
    temp=ComputeAffineMatrix(p1(D1(:,1),:),p2(D1(:,2),:)); %Using these S keypoints to calculate affine matrix.
    dists=ComputeError(temp,p1,p2,D2);%Calculate distance of keypoints between p1 and p2 after transform p2 using this affine matrix
    count=0; % Number of inliers except the S chosen points
    index=[];
    for i=1:size(dists,1)
        if dists(i)<maxInlierError
            count=count+1; %count the number of inliers
            index=[index;i];
        end
    end
    if count+seedSetSize>=goodFitThresh %If the number of inliers > goodFitThresh, using these inliers to refit the affine matrix
        M=[D1;D2(index,:)];
        temp=ComputeAffineMatrix(p1(M(:,1),:),p2(M(:,2),:));
        e=sum(ComputeError(temp,p1,p2,M));
        if e<n
            n=e; % n=minimum error
            H=temp; %H=affine matrix with minimum error
        end
    end
end
```

Parameter Adjustment

In my opinion, the most difficult of this project is parameter adjustment of RANSACFit. It would probably return eye(3) and 'No RANSAC fit was found' if you use the default parameters. There are 4 parameters that we could change to find a RANSAC fit or to make a better fit.

maxIter is the number of iterations, sometimes increasing it will find a RANSAC fit if you cannot find one, be it won't help if the maxIter is larger enough. Setting a large *maxIter* will make the program run longer but help little.

seedSetSize is the number of sample to compute affine matrix. Less *seedSetSize* will make it easier to find a RANSAC fit, but it will return "The matrix is close to a singular value, or the scaling is wrong. The results may be inaccurate." when compute affine matrix. But setting a small *seedSetSize* will help to find a RANSAC fit that could be used, even the result may not be good enough.

maxInlierError is the maximum distance between transformed point1 and point2. Larger *maxInlierError* will make RANSAC fit easier to find, but smaller *maxInlierError* will get a more accurate fit.

goodFitThresh is the threshold number for deciding whether the model is good or not. The larger *goodFitThresh* is, the more diffcult to find a RANSAC fit.

To find a RANSAC fit, first using default parameter and set a large *maxIter*. If still cannot find one, set a less *seedSetSize*. Then adjust *maxInlierError* to make it small enough, and make *goodFitThresh* large enough. After that, increase *seedSetSize* as long as a RANSAC fit could be find. Finally, decrease *maxIter* to make program faster.

How to choose image ?

Because we use affine tranformation, it cannot fit when images need perspective transformation, which means that we need to take pictures in a fixed direction, and try not to turn our camera's direction.

Furthermore, I found that it's easier to get a good fit when the resolution of the image is low, and it will be more accurate to stitch images with distant view than close objects.

Result