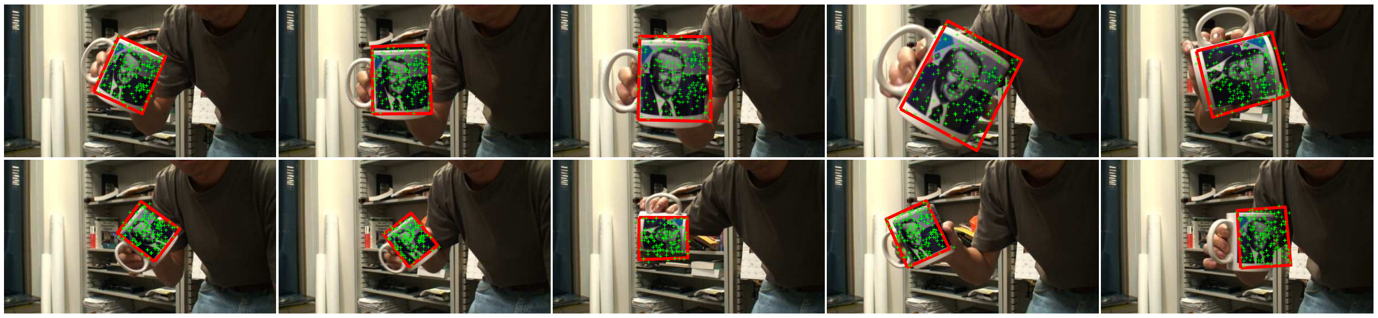
Model Fitting



In this assignment, we use RANSAC for object detection. You can use your images to test your code. You should write a report that includes codes and image processing results. Submit your report in PDF. The Matlab codes must also be submitted separately to blackboard. This assignment is due on Oct 22, 2011.

I. Object Localization Using RANSAC [60 points]

A. SIFT Feature

Based on the method we discuss in the class, implement the RANSAC method to detect object in images. The SIFT feature detector is attached with the assignment. The Matlab function `sift` wraps the SIFT executable to simplify the parameter manipulation. Make sure you copy both the Matlab code and the binary executables to your working directory. The code should work for both Windows and Mac. For more information about SIFT, check out David Lowe's SIFT demo web page at http://people.cs.ubc.ca/~lowe/keypoints/. SIFTPP web page is at http://www.vlfeat.org/~vedaldi/assets/siftpp/sift.html. SIFTPP is an implementation of David Lowe's SIFT which is accompanied with source code.

The interface of `sift` is

```
[image, descriptors, locs] = sift(imageFile)
%
% This function reads an image and returns its SIFT keypoints.
%   Input parameters:
%     imageFile: the file name for the image.
%
%   Returned:
%     image: the image array in double format
%     descriptors: a K-by-128 matrix, where each row gives an invariant
%         descriptor for one of the K keypoints.  The descriptor is a vector
%         of 128 values normalized to unit length.
%     locs: K-by-4 matrix, in which each row has the 4 values for a
%         keypoint location (row, column, scale, orientation).  The
%         orientation is in radians.
%   Credits: this code is adapted from David Lowe's SIFT demo and wraps SIFTPP and
%              siftWin32
```

Note that the SIFT feature locations have sub-pixel precisions.

A bunch of test images can also be downloaded from blackboard or the course web site. You can choose one image in the data set as the template and try to detect the selected object in other images.

B. RANSAC for Object Matching

You need to complete the following tasks:

1) Open a template image and select a region of interests, e.g. a cup, find the SIFT feature points in the region, the SIFT feature vectors and selected boundary points. The Matlab script is as follows

```
     im = imread(template_name);
```

```
im = im2double(im);
imwrite(im, ’tmp.pgm’); %sift only accepts pgm
imshow(im);

[mask, xc, yc] = roipoly(im);
mask = im2double(mask);
save mask.mat mask

% Compute the sift features in the template image
[image1, des1, loc] = sift(’tmp.pgm’);

% Select sift in the region of interest
z = [];
ff = [];
for n = 1 : size(loc,1)
    r = floor(loc(n,2));
    c = floor(loc(n,1));
    if r > 0 & c > 0 & mask(r, c) == 1
        z = [z; floor(loc(n,1:2))];
        ff = [ff; des1(n,:)];
    end
end
hold on;
plot(z(:,1), z(:,2), ’+’);

p_src = unique(z, ’rows’); % the sift points in ROI
[ism, id] = ismember(p_src, z, ’rows’);
feature1 = ff(id,:); % the sift features in ROI
```

At the end of the script, you will have sift locations in `p_src` and features in `feature1`.

2) Open a target image, compute the SIFT features and compare the similarity of each template target feature pairs. We then use a heuristic to find the promising matching pairs. In the heuristic, we keep only the matching pairs whose best matching costs are below some percentage of the second best matching costs.

```
im = imread(target_image);
imwrite(im, ’tmp.pgm’);
[image2, feature2, loc2] = sift(’tmp.pgm’);

%Compute the similarity of the template with the target sift points
feature2 = feature2’;
dotprods = feature1 * feature2;        % Computes vector of dot products
cc = acos(dotprods)/pi*2;  % the matching costs

p_des = loc2(:,1:2); % the target point locations

% Find some promising point-point pairs
dx = [];
dy = [];
for n = 1 : size(p_src,1)
    [vals,indx] = sort(cc(n,:));
    if (vals(1) < 0.85*vals(2))
       dx = [dx; p_src(n,1), p_src(n,2), p_des(indx(1),1)];
       dy = [dy; p_src(n,1), p_src(n,2), p_des(indx(1),2)];
    end
end
```

In the above procedure, `dx` contains the source points x and y in the first two columns and the target x in the third column; `dy` contains the source points x and y in the first two columns and the target y in the third column.

3) Use RANSAC to align the template points with the target points. You need to randomly pick up 3 template-target point pairs using `dx` and `dy`. You can use the following function to generate 3 different random numbers.

```
function r3 = rand3(N)
% rand3 randomly generates 3 different numbers in 1 to n.
% r3 contains the the 3 random numbers
   t = 1 : N;
   for n = 1 : 3
       r = floor(rand(1,1)*(N-n+1)) + n;
       tmp = t(n);
       t(n) = t(r);
       t(r) = tmp;
   end
   r3 = t(1:3);
```

Assuming that you have got 3 random numbers in r3, the transformation from the template to the target can be computed using the Matlab function

```
t = cp2tform(in_points, out_points, 'similarity');
```

where `in_points=[dx(r3,1:2)]` and `out_points=[dx(r3,3) dy(r3,3)]`. You can then project the template points to the target image using,

```
p_proj = [p_src, ones(size(p_src,1), 1)] * t.tdata.T;
```

The first two columns of `p_proj` is the x and y coordinates of the projected points. You can also use `'affine'` in the function of `cp2tform`. After mapping each template point to the target image, compare the feature with the nearest target point and keep the overall matching cost. Repeat this procedure for N, .e.g. 100 times, and pick up the one with the minimum overall cost. Project the template points to the target image, you locate the target object.

4) Test your code. If you use your own test images, you should also include them in your submission.

II. Skeleton Code

The complete skeleton code is as follow.

```
% Read the template image and select a region of interest
template_name = './images/template.jpg';

im = imread(template_name);
im = im2double(im);
imwrite(im, 'tmp.pgm');
imshow(im);

[mask, xc, yc] = roipoly(im);
mask = im2double(mask);
save mask.mat mask

% Compute the sift features in the template image
[image1, des1, loc] = sift('tmp.pgm');

% Select sift in the region of interest
z = [];
ff = [];
for n = 1 : size(loc,1)
    r = floor(loc(n,2));
    c = floor(loc(n,1));
    if r > 0 & c > 0 & mask(r, c) == 1
        z = [z; floor(loc(n,1:2))];
        ff = [ff; des1(n,:)];
    end
end
hold on;
plot(z(:,1), z(:,2), '+');

p_src = unique(z, 'rows'); % the sift points in ROI
[ism, id] = ismember(p_src, z, 'rows');
```

```
feature1 = ff(id,:); % the sift features in ROI

%------A big loop ---------%
for videon = 1 : 40
     videon
     fname2 = sprintf('./images/target%d.jpg', videon);
     im = imread(fname2);
     imwrite(im, 'tmp.pgm');
     [image2, feature2, loc2] = sift('tmp.pgm');

     %Compute the similarity of the template with the target sift points
     feature2 = feature2';
     dotprods = feature1 * feature2;        % Computes vector of dot products
     cc = acos(dotprods)/pi*2;  % the matching costs

     p_des = loc2(:,1:2); % the target point locations

     % Find some promising point-point pairs
     dx = [];
     dy = [];
     for n = 1 : size(p_src,1)
         [vals,indx] = sort(cc(n,:));
         if (vals(1) < 0.85*vals(2))
            dx = [dx; p_src(n,1), p_src(n,2), p_des(indx(1),1)];
            dy = [dy; p_src(n,1), p_src(n,2), p_des(indx(1),2)];
         end
     end


     %--- ransac ---------------%

         Here is your ransac code

     %-------------------------%

         Show the results

     %--------------------------%
end
```