16-720J: Homework 4
Tracking Templates and Control Points

Luting Wang

November 12, 2015

1 The Car Tracker: Template Tracking with Lucas-Kanade (10 pts)

Question 1.1 Warmup (5pts)

$$A^T A = \begin{bmatrix} \Sigma I_x^2 & \Sigma I_x I_y \\ \Sigma I_x I_y & \Sigma I_y^2 \end{bmatrix} \tag{1}$$

The matrix $A^T A$ is same as Harris Matrix.

It should be invertible($A^T A \neq 0$),i.e. no zero eigenvalues, so that the template offset can be calculated reliably.

Question 1.2 Discussion (5pts)

It runs smoothly but slow. And it may fail on the following scenarios: First, it may fail if the car moves very fast or it is shielded by some other objects. It may also fail if lighting is not constant or there is a viewpoint change of the tracked car. To remedy such problems, we can use multiple components of the car to track, or to apply SDM to it.

2 The Pooh Tracker: Component-based Tracking (90 pts)

2.1 Tracking the Pooh with the LK Tracker

Question 2.1.1 Implementation (8 pts)

```
% Created by wanglut (NOV-5-2015)

addpaths;

load('data/pooh/rects_frm992.mat');
Files = dir(fullfile('data/pooh/testing','*.jpg'));
LengthFiles = length(Files);

for i = 1:LengthFiles
    fprintf('loading the image %s\n', Files(i).name);
    sequence(:,:,:,i) = imread(strcat('data/pooh/testing/',Files(i).name));
end

vidname = 'pooh_lk.avi';
vidout  = VideoWriter(vidname);
vidout.FrameRate = 10;
```

```
open(vidout);

rect = {rect_lear, rect_leye, rect_nose, rect_rear, rect_reye};
drawFrmPooh(sequence, rect,1); ...
    text(80,100,'Ready?','color','r','fontsize',30); pause(1);
drawFrmPooh(sequence, rect,1); ...
    text(80,160,'GO!','color','g','fontsize',80); pause(.5);

for iFrm = 2:LengthFiles
    It    = sequence(:,:,:,iFrm-1);    % get previous frame
    It1   = sequence(:,:,:,iFrm);      % get current frame
    for j = 1:5
        [u,v] = LucasKanade(It,It1,rect{j}); % compute the displacement ...
            using LK
        rect{j}  = rect{j} + [u,v,u,v];          % move the rectangle
    end
    hf  = drawFrmPooh(sequence, rect, iFrm); % draw frame
    frm = getframe;
    writeVideo(vidout, imresize(frm.cdata, 0.5));
end

close(vidout);
close(1);
fprintf('Video saved to %s\n', vidname);
```

Question 2.1.2 Discussions (2 pts)

I can't track until 3000. When it is in the frame 1782, the left ear lost track. As you can see below.I think the reason it fails is because there is a viewpoint change.

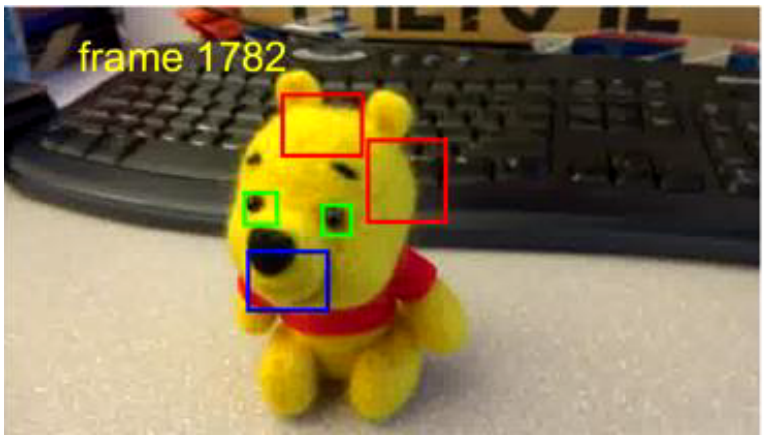

Figure 1: My tracker got lost on frame number 1782

Question 2.1.3 Extra Credit (10 pts)

There are two major modifications I made. First, in original LK, when it is in the Frame 1510, it stops working because of infinite iteration inside LK (because it starts from a saddle point, it cannot converge). So I add iteration counter inside the function, and if the number of iteration is above 200, it jumps of out the loop. Therefore, it can keep running until 3000 frame. Also, in original LK, pooh's right eye loses track when in frame 1276. I think it may because the initial rectangle for right eye is not appropriate. So I modify its rect_reye's topLeftY from 239 to 234, and bottomRightY from 271 to 267, then it can track until 1782 when there is a viewpoint change.

2.2 Tracking the Pooh with Supervised Descent Method (SDM) Tracker

Question 2.2.1 Training step 1: perturbation (10 pts)

```
function perturbedConfigurations = ...
    genPerturbedConfigurations(singleFrameAnnotation, meanShape, n, ...
    scalesToPerturb)

scale = findscale(singleFrameAnnotation, meanShape);
meanShape = meanShape/scale;
[center_shift] = mean(singleFrameAnnotation) -mean(meanShape);
meanShape(:,1) = meanShape(:,1)+ center_shift(1);
meanShape(:,2) = meanShape(:,2)+ center_shift(2);

perturbedConfigurations = zeros(4, 5*n);
[center_pos] = mean(meanShape);
for i = 1:n
    scale_to_pertubred = scalesToPerturb(ceil(size(scalesToPerturb, ...
        2)*rand));
    scale_meanShape = scale_to_pertubred*meanShape;
    scale_center = mean(scale_meanShape);
    certer_shiftx = center_pos(1) - scale_center(1);
    certer_shifty = center_pos(2) - scale_center(2);

    scale_meanShape(:,1) = scale_meanShape(:,1)+ certer_shiftx;
    scale_meanShape(:,2) = scale_meanShape(:,2)+ certer_shifty;

    x_move = (rand*2-1)*5;
    y_move = (rand*2-1)*5;

    for j = 1:5
        perturbedConfigurations(1, (i-1)*5+j) = scale_meanShape(j, 1)+ ...
            x_move;
        perturbedConfigurations(2, (i-1)*5+j) = scale_meanShape(j, 2)+ ...
            y_move;
    end

    perturbedConfigurations(3, (i-1)*5+1:i*5) = [7 4 4 10 ...
        10]/scale*scale_to_pertubred;
end

end
```

Question 2.2.2 Training steps 2&3: prepare D and F (10 pts)

```
function D = genDisplacementMatrix(perturbedConfigurations,annotations, n)

numOfFrame = length(perturbedConfigurations);
D = zeros(numOfFrame*n, 10);

for i = 1:numOfFrame
    perturbedPositon = reshape(perturbedConfigurations{i}(1:2, :), [1, ...
        n*10]);
    D((i-1)*n+1:i*n,:) = (reshape(repmat(annotations(i, 2:11), 1, ...
        n)-perturbedPositon,[10, n]))';
end
```

```
function F = genFeatureMatrix(perturbedConfigurations, n)

Files = dir(fullfile('data/pooh/training','*.jpg'));
numOfFrame = length(perturbedConfigurations);
F = zeros(numOfFrame*n, 640);

for i = 1:numOfFrame
     I = imread(strcat('data/pooh/training/',Files(i).name));
     total_sift = siftwrapper(I, perturbedConfigurations{i});
     for j = 1:n
         sift = total_sift(:, (j-1)*5+1:j*5);
         for m = 1:5
            norm_sift = norm(sift(:, m));
            sift = sift./norm_sift;
         end
         F((i-1)*n+j, :) = sift(:)';
     end
end
```

Question 2.2.3 Training steps 4&5: linear mapping and update configuration (10 pts)

```
function [new_perturbedConfigurations, models] ...
   =learnMappingAndUpdateConfiguration(F, D, perturbedConfigurations, n, ...
   annotations)

    models = learnLS(F, D);
    new_displacement = F*models;

    numOfFrame = length(perturbedConfigurations);

    for i = 1:numOfFrame
        perturbedPositon = reshape(perturbedConfigurations{i}(1:2, :), ...
            [1, n*10]);
        new_position = new_displacement((i-1)*n+1:i*n, :) + ...
            reshape(perturbedPositon, 10, n)';
       temp = reshape(new_position(1, :), [2, 5]);
       singleFrameAnnotation = reshape(annotations(i, 2:11), [2, 5])';
       scale = findscale(singleFrameAnnotation, temp');
       new_perturbedConfigurations{i}(3, 1:5) = [7 4 4 10 10]/scale;
       for j = 2:n
           scale = findscale(singleFrameAnnotation, ...
               reshape(new_position(j, :), [2, 5])');
           temp = [temp,reshape(new_position(j, :), [2, 5])];
           new_perturbedConfigurations{i}(3, 5*(j-1)+1:5*j) = [7 4 4 10 ...
               10]/scale;
       end
       new_perturbedConfigurations{i}(1:2, :) = temp;
       new_perturbedConfigurations{i}(4, :) = zeros(1, n*5);
    end

end
```

Question 2.2.4 Sequentially learn multiple mappings (10 pts)

In this section, I got 5 mapping function. And to validate its correctness, I checked the loss function. In order to do that, I apply norm on the displacement and see its change. After

5 updated displacements, value of norm is monotonously decreased from 70 to 10, which is reasonable. Please refer to SDMtrain.m in the next section.

Question 2.2.5 Integration (15 pts)

```
function [models, loss] = SDMtrain(mean_shape, annotations)

n = 100;
scalesToPerturb = [0.8, 1.0, 1.2];

numOfFrame = size(annotations, 1);
perturbedConfigurations = cell(numOfFrame, 1);

for i = 1:numOfFrame
     singleFrameAnnotation = reshape(annotations(i, 2:11), [2, 5])';
     perturbedConfigurations{i} = ...
         genPerturbedConfigurations(singleFrameAnnotation, mean_shape, n, ...
         scalesToPerturb);
end

D = genDisplacementMatrix(perturbedConfigurations,annotations, n);
F = genFeatureMatrix(perturbedConfigurations, n);

models = cell(10, 1);
loss = zeros(1, 10);

for i = 1:10
    [new_perturbedConfigurations, models{i}] = ...
        learnMappingAndUpdateConfiguration(F, D, perturbedConfigurations, ...
        n, annotations);
    D = genDisplacementMatrix(new_perturbedConfigurations, annotations, n);
    loss(i) = norm(D);
    F = genFeatureMatrix(new_perturbedConfigurations, n);
    perturbedConfigurations = new_perturbedConfigurations;
end
```

Question 2.2.6 Implementation (20 pts)

```
function SDMtrack(models, mean_shape, start_location, start_frame, ...
   outvidfile)

    vidout = VideoWriter(outvidfile);
    vidout.FrameRate = 20;
    open(vidout);

    new_Configurations= zeros(4, 5);
    numOfModel = length(models);
    for iFrm = start_frame:3000

       I = imread(sprintf('data/pooh/testing/image-%04d.jpg', iFrm));
       scale = findscale(start_location, mean_shape);
       mean_shape = mean_shape/scale;
       center_shift = mean(start_location-mean_shape);
       mean_shape(:,1) = mean_shape(:,1)+ center_shift(1);
       mean_shape(:,2) = mean_shape(:,2)+ center_shift(2);

       begin_shape =  mean_shape;
```

```
        current_shape = begin_shape;
        new_Configurations(1:2, :) = begin_shape';
        new_Configurations(3, :) = [7 4 4 10 10]/scale;

         for i = 1:numOfModel
             temp_sift = siftwrapper(I, new_Configurations);
                for m = 1:5
                     sift_norm = norm(temp_sift(:, m));
                     temp_sift(:, m) = temp_sift(:, m)/sift_norm;
                end
                sift = temp_sift(:)';
                new_displacement = sift*models{i};

               current_shape = current_shape + reshape(new_displacement, ...
                  [2, 5])';
               scale = findscale(start_location, current_shape);
               new_Configurations(1:2, :) = current_shape';
               new_Configurations(3, :) = [7 4 4 10 10]/scale;
         end

         figure(1);
         if ¬exist('hh','var'), hh = imshow(I); hold on;
         else set(hh,'cdata',I);
         end
         if ¬exist('hPtBeg','var'), hPtBeg = plot(begin_shape(:,1), ...
            begin_shape(:,2), 'g+', 'MarkerSize', 15, 'LineWidth', 3);
         else set(hPtBeg,'xdata',begin_shape(:,1),'ydata',begin_shape(:,2));
         end
         if ¬exist('hPtcurrent_shape','var'), hPtcurrent_shape = ...
            plot(current_shape(:,1), current_shape(:,2), 'r+', ...
            'MarkerSize', 25, 'LineWidth', 5);
         else ...
            set(hPtcurrent_shape,'xdata',current_shape(:,1),'ydata',current_shape(:,2));
         end
         title(['frame ',num2str(iFrm)]);
         if ¬exist('hFrmNum', 'var'), hFrmNum = text(30, 30, ['Frame: ...
            ',num2str(iFrm)], 'fontsize', 40, 'color', 'r');
         else set(hFrmNum, 'string', ['Frame: ',num2str(iFrm)]);
         end

         frm = getframe;
         writeVideo(vidout, imresize(frm.cdata, 0.5));

         start_location = current_shape;
     end

     close(vidout);
end
```

Question 2.2.7 Discussions (5 pts)

I think the second one, SDM tracker, is better. The advantages of it are following: First, it is more robust since we assume a statistical motion model on it (meanshape). Second, it is able to re-track the object once get lost, because it uses the sift feature instead of previous frame to check.

The disadvantages are: it requires training beforehand, which need training set. Second, during training, it is very computationally expensive. Because we try to learn a series of descent directions and re-scaling factors such that it produces a sequence of updates.

And the advantages of the first algorithm are that it is easy to implement and don't need

training. Also, it provides more accurate estimation than SDM when converges, because SDM uses a generic descent direction.

And the disadvantages are it is not robust against bad initializations or ill-conditions, since it is based on two frames. And it need to compute the Jacobian or the Hessian when tracking, which influence tracking speed.

Question 2.2.8 Extra credits (3+3+3 pts)

The first challenge starts at the frame 1410, and my tracker succeeds in tracking even if there is a big scale change. The reason is that I use multiple scaled perturbation when training(0.8, 1.0, 1.2). So during testing, if there is a scale change, it will not lose track. And for the final two challenges, to recover them, I made some modifications. First, I change the scalesToPerturb configuration(0.5, 0.8, 1.0, 1.2, 1.5, 2.0), and then I double the map model to 10. But the effect is not obvious. But for the final challenge, it succeeds to recover it. And at last, when the scale is normal, it can re-track the Winnie.Please refer to Figure 2, 3, 4, or pooh_sdm.avi.

Figure 2: First challenge when there is a big scale change

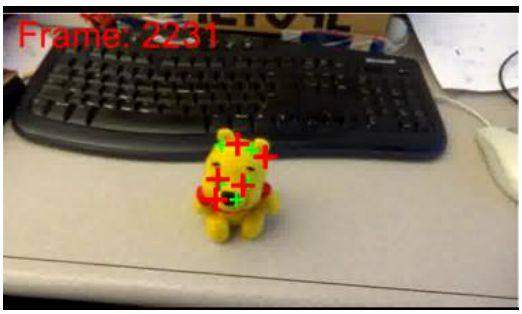

Figure 3: Second challenge when it becomes super small

Figure 4: Third challenge when there is slightly bigger pose change

Question 2.2.9 Extra credits (10 pts) I didn't attempt this part.