Amath 482 Hw 3 - Oscillating Can: Single Value Decomposition

Ofir Kedar

February 24, 2021

Abstract

The purpose of this project is to utilize Principal Component Analysis (PCA) through the implementation of Single Value Decomposition (SVD) to analyze the implications they have on systems. We see that in a swinging rotating and oscillating pendulum as more principal components and noise are added to a system the more ranks of the original data matrix are needed to model the system accurately.

1 Introduction and Overview

Given video footage of a mass on a spring, we are asked to apply PCA using SVD to see what variations on the mass's motion and camera-work quality, do to the ability to estimate and model the situation using a rank 1 through rank 6 (full rank) approximation. We are given 4 situations of a can in motion. Case 1 - Ideal - where the can is essentially only moving in the y direction (up and down). Case 2 - Noisy - repeats Case 1 but adds camera shaking. Case 3 - Horizontal Displacement - where the mass is released off center creating motion in the x-z plane. Case 4 - repeats case 3 but the can is also rotating about its center axis is parallel to the y direction. Each situation is filmed from 3 angles, where rank 1 contains only the information from x or y of 1 of the 3 cameras, rank 2 uses x and y of the one or two cameras that tell s the most, and so on.

2 Theoretical Background

2.1 Single Value Decomposition (SVD)

Single Value Decomposition gives us information about a matrix in the form of 3 matrices, of of them tell us about the rotation/ orientation of the matrix, and one tells us about its stretching in those directions. We have:

$$\text{Single Value Decomposition} : A = U\Sigma V^* \tag{1}$$

Here, $\boldsymbol{A}$ is a matrix containing out original data. $\boldsymbol{U}$ is one of the resulting matrices that accounts for rotation and contains the time series data. $\boldsymbol{V^*}$ is the other rotation matrix, which when inverted to solve for matrix V gives the principal components. Matrix $\boldsymbol{\Sigma}$ is the stretching matrix and gives the single value energies, which are used to determine the percent of information about the system contained in each rank of matrix A. [1]

2.2 Principal Component Analysis (PCA)

The over arching idea in PCA is to discard correlations between different variables in a given data set. For example although there is an upward correlation between a persons height and weight on a plot, we want to reorient our bearings such that as you move along in the x direction, you are not given any information bout the y direction, and have a mean of 0. This allows us to analyze each component separately and can tell us how much information is contained in each component in relation to the whole system. [1]

$$\text{Coriance of A} : C_X = \frac{1}{n-1} X X^T \tag{2}$$

3 Algorithm Implementation and Development

3.1 Motion Capture

In order to begin analysis PCA on the motion of our can, we must first extract the motion data from our videos. Conveniently, the parts of the can are bright white, and there is a while light on the corner of the can in many of the videos. This means that in a gray color scale of each frame in the video, the can should contain the max valued pixel in the frame, or at least a very high value. This means we can track the location of the max value in the matrix containing the image at each frame in the video and through this track the motion of the can. To ensure that no other light or white background is confused for the location of the can, we can manually find the cans original location by finding a generically bright pixel on it, and create a window around it. Everything around the window of the next frame can now be set to equal 0, since the can won't just appear in a random spot on that frame. Now when we find the maximum it is actually just a local maximum, and will follow only the white in the small window allotted which will contain the can in the next frame.

Algorithm 1: Motion Capture

```
load('cam1_2.mat')
GrayFrameAll = vidFrames1_2(:,:,:,13:end);
numFrames = size(GrayFrameAll,4);
X_time1_2 = zeros(1,numFrames);
Y_time1_2 = zeros(1,numFrames);

for j = 1:numFrames do
   GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));
   % x filter
   GrayFrame(:,1:x-50) = 0;
   GrayFrame(:,x+50:end) = 0;
   % y filter
   GrayFrame(1:y-50,:) = 0;
   GrayFrame(y+20:end,:) = 0;

   [Max, index] = max(GrayFrame(:));
   y,x= ind2sub(size(GrayFrame), index);
   X_time1_2(j) = x;
   Y_time1_2(j) = y;
end for
S2(1,:) = X_time1_2;
S2(2,:) = Y_time1_2;
```

3.2 Principal Component Analysis

In order to compute the energies of our principal components we run SVD on the scaled Matrix A containing the X Y coordinates over time in each video. This is done in one command function in Matlab **[U,S,V] = svd(A/sqrt(n-1), 'econ')**, where recalling from section 2.1, U = U, S = Σ, and V = V.

$$\mathrm{A} = \begin{pmatrix} x\_11 & x\_12 & .... & x\_1n \\ y\_11 & y\_12 & .... & y\_1n \\ x\_21 & x\_22 & .... & x\_2n \\ y\_21 & y\_22 & .... & y\_2n \\ x\_31 & x\_32 & .... & x\_3n \\ y\_31 & y\_32 & .... & y\_3n \end{pmatrix}$$

4 Computational Results

4.1 Graphical Evaluation of All Trial Cases

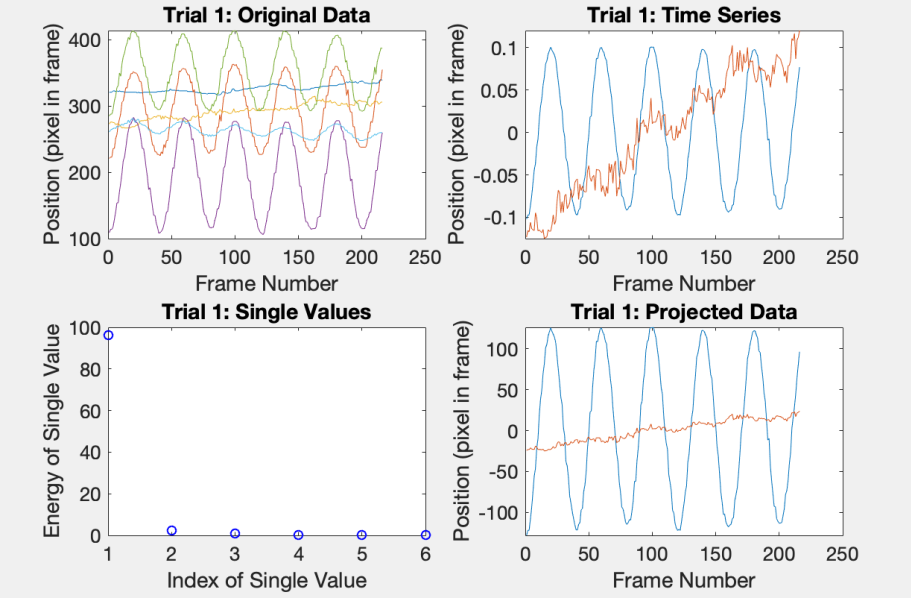

Figure 1: Trial Case 1

Figure 2: Trial Case 2

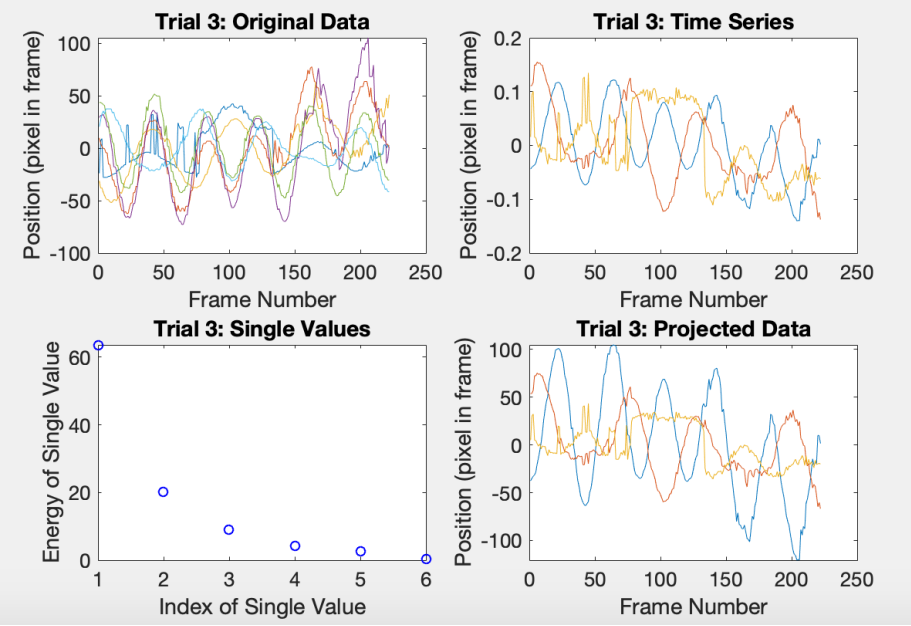

Figure 3: Trial Case 3

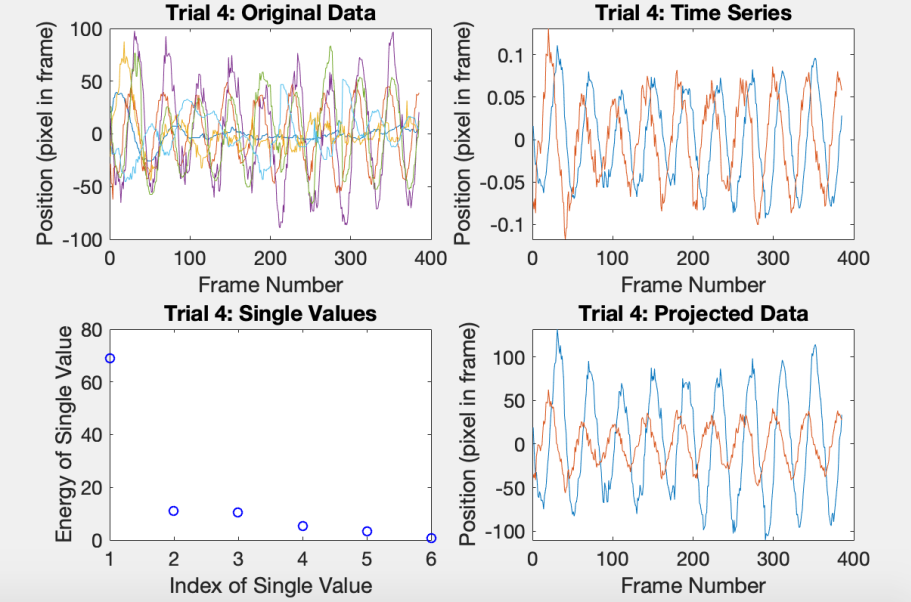

Figure 4: Trial Case 4

4.2 Original Data Graph Break Down

Each color line represents a directional array of positional data from our original matrix A as shown in section 3.2. Looking at Trial 1 is the best way to interpret this data as it is the closest we have to an ideal situation. We can see the Y components oscillating of with a cos(time) like function of a large magnitude, and x components are fairly linear and close relatively parallel to the time axis. This is exactly what we observe in the trial 1 footage. Things become much less harder to decipher as we introduce noise to this system in trial 2 and 4, but in trial 3 the oscillatory like behavior can somewhat be seen again in the y and

in the x direction as well - given that we already know this is what's happening in the videos.

4.3 Time Series Graph Break Down

V contains the time series, where the magnitude of our modes over time is displayed, we chose to plot those that associate with single values that are high enough to imply relevance of the data.

4.4 Single Value Graph Break Down

This graph is arguably the most important and informative one we can obtain. By graphing the σ^2 values on a percentile graph, we extract the quantitative analysis of the amount of relevant information stored in in each rank of our SVD. trial 1 we see that essentially all information about the motion of the can is captured by a single axis of a single camera. This makes sense because baring very minimal noise due to human influence, we only observe vertical motion. Although the same is true about the can in trial 2, there is a lot of noise, creating horizontal motion from the perspective of all 3 cameras. There is now more variance in position of the can in the x directions of the footage from each camera, or in other words, the total amount of relevant information has increased, but the amount of information about the y location is very similar to before, so the percentage of its relevance from a single camera has reduced. We see the rank 1 single value go down to around 70% of the information about our system, as apposed to its roughly 96% from before in trial 1.

4.5 Projected Data Graph Break Down

Matrix U gives us our principal components. This means that if we project our transposed original data shifted about the mean onto matrix U we will see the data approximation from each mode. Again we look for a relevance of principal components by looking to their associated single values and their magnitude in relation to the other single values.

5 Summary and Conclusions

Single Value Decomposition is a powerful tool, and Principal Component Analysis is one of the ways we can use it to extract meaningful information large data sets. In order to maximize efficiency of data we are able to utilise PCA to determine relevant portions of a large data set. By extracting the singular values we can tell how relevant these data set portions are and choose to what precision we would like to approximate different systems. Using differently ranked approximations we can then model these systems to a desired precision, allowing us to discard or include components.

References

[1] Jose Nathan Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data*. Oxford University Press, 2013.

Appendix A MATLAB Functions

Add your important MATLAB functions here with a brief implementation explanation. This is how to make an **unordered** list:

- `y = linspace(x1,x2,n)` returns a row vector of `n` evenly spaced points between `x1` and `x2`.
- `[X,Y,Z] = meshgrid(x,y,z)` returns 3-D grid coordinates defined by the vectors x, y, and z. The grid represented by X, Y, and Z has size length(y)-by-length(x)-by-length(z).
- [Max_ave, Max_index] = max(abs(ave(:))) returns the max absolute value in matrix ave and it's index location
- [I1,I2,I3,...,In] = ind2sub(SIZ,IND) returns N subscript arrays I1,I2,..,In containing the equivalent N-D array subscripts equivalent to IND for an array of size SIZ.
- I = rgb2gray(RGB) converts the truecolor image RGB to the grayscale intensity image I.
- implay('filename') opens the implay movie player, displaying the content of the file specified by 'filename'.
- imshow(I) displays the grayscale image I
- B = repmat(A) creates a large matrix B the same size as A, tiling of copies of A. If A is a matrix, the size of B is [size(A,1)*M, size(A,2)*N].
- [U,S,V] = svd(X) produces a diagonal matrix S, of the same dimension as X and with non-negative diagonal elements in decreasing order, and unitary matrices U and V so that X = U*S*V'.

Appendix B MATLAB Code

```
% clear all;  close all;  clc;



%%                                      Tracking
%%
%                                      cam1_1                              1

% close all
load('cam1_1.mat')
% implay(vidFrames1_1)

GrayFrameAll = vidFrames1_1(:,:,:,11:226);
numFrames = size(GrayFrameAll,4);
X_time1_1 = zeros(1,numFrames);
Y_time1_1 = zeros(1,numFrames);

% initial max
x = 323;
y = 222;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+50:end) = 0;
    % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+20:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time1_1(j) = x;
    Y_time1_1(j) = y;

     figure(1);
     imshow(GrayFrame);
     hold on
     plot(X_time1_1(j), Y_time1_1(j), 'bo', 'markersize', 10);
     drawnow
end

S1(1,:) = X_time1_1;
S1(2,:) = Y_time1_1;
```

Listing 1:

```
%%
%                                   cam2_1                              1

% close all
load('cam2_1.mat')
% implay(vidFrames2_1)

GrayFrameAll = vidFrames2_1(:,:,:,20:20+215);
numFrames = size(GrayFrameAll,4);
X_time2_1 = zeros(1,numFrames);
Y_time2_1 = zeros(1,numFrames);

% initial max
x = 277;
y = 111;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+50:end) = 0;
    % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+20:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time2_1(j) = x;
    Y_time2_1(j) = y;

%     figure(2);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time2_1(j), Y_time2_1(j), 'bo', 'markersize', 10);
%     drawnow
end

S1(3,:) = X_time2_1;
S1(4,:) = Y_time2_1;
```

Listing 2:

```
%%
%                                   cam3_1                              1

% close all
load('cam3_1.mat')
% implay(vidFrames3_1)

GrayFrameAll = (vidFrames3_1(:,:,:,11:11+215));
numFrames = size(GrayFrameAll,4);
X_time3_1 = zeros(1,numFrames);
Y_time3_1 = zeros(1,numFrames);

% initial max
x = 286;
y = 262;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

        % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+50:end,:) = 0;
        % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+20:end) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time3_1(j) = x;
    Y_time3_1(j) = y;

%     figure(3);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time3_1(j), Y_time3_1(j), 'bo', 'markersize', 10);
%     drawnow
end

S1(5,:) = X_time3_1;
S1(6,:) = Y_time3_1;
```

Listing 3:

```
%%
%                                   cam1_2                          2

% close all
load('cam1_2.mat')
% implay(vidFrames1_2)

GrayFrameAll = vidFrames1_2(:,:,:,13:end);
numFrames = size(GrayFrameAll,4);
X_time1_2 = zeros(1,numFrames);
Y_time1_2 = zeros(1,numFrames);

% initial max
x = 332;
y = 332;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+50:end) = 0;
    % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+20:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time1_2(j) = x;
    Y_time1_2(j) = y;

%     figure(1);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time1_2(j), Y_time1_2(j), 'bo', 'markersize', 10);
%     drawnow
end

S2(1,:) = X_time1_2;
S2(2,:) = Y_time1_2;
```

Listing 4:

```
%%
%                                   cam2_2                          2

% close all
load('cam2_2.mat')
% implay(vidFrames2_2)

GrayFrameAll = vidFrames2_2(:,:,:,1:1+301);
numFrames = size(GrayFrameAll,4);
X_time2_2 = zeros(1,numFrames);
Y_time2_2 = zeros(1,numFrames);

% initial max
x = 317;
y = 360;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+50:end) = 0;
    % y filter
    GrayFrame(1:y-75,:) = 0;
    GrayFrame(y+50:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time2_2(j) = x;
    Y_time2_2(j) = y;

%     figure(1);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time2_2(j), Y_time2_2(j), 'bo', 'markersize', 10);
%     drawnow
end

S2(3,:) = X_time2_2;
S2(4,:) = Y_time2_2;
```

Listing 5:

```
%%
%                                   cam3_2                          2

% close all
load('cam3_2.mat')
% implay(vidFrames3_2)

GrayFrameAll = vidFrames3_2(:,:,:,18:18+301);
numFrames = size(GrayFrameAll,4);
X_time3_2 = zeros(1,numFrames);
Y_time3_2 = zeros(1,numFrames);

% initial max
x = 412;
y = 247;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-70) = 0;
    GrayFrame(:,x+30:end) = 0;
    % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+50:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time3_2(j) = x;
    Y_time3_2(j) = y;

%     figure(1);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time3_2(j), Y_time3_2(j), 'bo', 'markersize', 10);
%     drawnow
end

S2(5,:) = X_time3_2;
S2(6,:) = Y_time3_2;
```

Listing 6:

```
%%
%                                   cam1_3                              3

%close all
load('cam1_3.mat')
% implay(vidFrames1_3)

GrayFrameAll = vidFrames1_3(:,:,:,18:18+221);
numFrames = size(GrayFrameAll,4);
X_time1_3 = zeros(1,numFrames);
Y_time1_3 = zeros(1,numFrames);

% initial max
x = 301;
y = 317;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+50:end) = 0;
    % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+20:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time1_3(j) = x;
    Y_time1_3(j) = y;

%     figure(1);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time1_3(j), Y_time1_3(j), 'bo', 'markersize', 10);
%     drawnow
end

S3(1,:) = X_time1_3;
S3(2,:) = Y_time1_3;
```

Listing 7:

```
%%
%                                   cam2_3                              3

% close all
load('cam2_3.mat')
% implay(vidFrames2_3)

GrayFrameAll = vidFrames2_3(:,:,:,44:44+221);
numFrames = size(GrayFrameAll,4);
X_time2_3 = zeros(1,numFrames);
Y_time2_3 = zeros(1,numFrames);

% initial max
x = 267;
y = 295;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-20) = 0;
    GrayFrame(:,x+20:end) = 0;
    % y filter
    GrayFrame(1:y-60,:) = 0;
    GrayFrame(y+15:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time2_3(j) = x;
    Y_time2_3(j) = y;

%     figure(1);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time2_3(j), Y_time2_3(j), 'bo', 'markersize', 10);
%     drawnow
end

S3(3,:) = X_time2_3;
S3(4,:) = Y_time2_3;
```

Listing 8:

```
%%
%                                   cam3_3                              3

% close all
load('cam3_3.mat')
% implay(vidFrames3_3)

GrayFrameAll = vidFrames3_3(:,:,:,9:9+221);
numFrames = size(GrayFrameAll,4);
X_time3_3 = zeros(1,numFrames);
Y_time3_3 = zeros(1,numFrames);

% initial max
x = 379;
y = 272;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-70) = 0;
    GrayFrame(:,x+30:end) = 0;
    % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+50:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time3_3(j) = x;
    Y_time3_3(j) = y;

%     figure(1);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time3_3(j), Y_time3_3(j), 'bo', 'markersize', 10);
%     drawnow
end

S3(5,:) = X_time3_3;
S3(6,:) = Y_time3_3;
```

Listing 9:

```
%%
%                                   cam1_4                              4

% close all
load('cam1_4.mat')
% implay(vidFrames1_4)

GrayFrameAll = (vidFrames1_4(:,:,:,8:8+384));
numFrames = size(GrayFrameAll,4);
X_time1_4 = zeros(1,numFrames);
Y_time1_4 = zeros(1,numFrames);

% initial max
x = 401;
y = 317;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+50:end) = 0;
    % y filter
    GrayFrame(1:y-20,:) = 0;
    GrayFrame(y+50:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time1_4(j) = x;
    Y_time1_4(j) = y;

%     figure(3);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time1_4(j), Y_time1_4(j), 'bo', 'markersize', 10);
%     drawnow
end

S4(1,:) = X_time1_4;
S4(2,:) = Y_time1_4;
```

Listing 10:

```
%%
%                                    cam2_4                              4

% close all
load('cam2_4.mat')
% implay(vidFrames2_4)

GrayFrameAll = (vidFrames2_4(:,:,:,8:8+384));
numFrames = size(GrayFrameAll,4);
X_time2_4 = zeros(1,numFrames);
Y_time2_4 = zeros(1,numFrames);

% initial max
x = 280;
y = 272;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+50:end) = 0;
    % y filter
    GrayFrame(1:y-20,:) = 0;
    GrayFrame(y+50:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time2_4(j) = x;
    Y_time2_4(j) = y;

%     figure(3);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time2_4(j), Y_time2_4(j), 'bo', 'markersize', 10);
%     drawnow
end

S4(3,:) = X_time2_4;
S4(4,:) = Y_time2_4;
```

Listing 11:

```
%%
%                                   cam3_4                          4

% close all
load('cam3_4.mat')
% implay(vidFrames3_4)

GrayFrameAll = (vidFrames3_4(:,:,:,1:1+384));
numFrames = size(GrayFrameAll,4);
X_time3_4 = zeros(1,numFrames);
Y_time3_4 = zeros(1,numFrames);

% initial max
x = 414;
y = 203;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

        % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+50:end,:) = 0;
        % x filter
    GrayFrame(:,1:x-50) = 0;
    GrayFrame(:,x+20:end) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time3_4(j) = x;
    Y_time3_4(j) = y;

%     figure(3);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time3_4(j), Y_time3_4(j), 'bo', 'markersize', 10);
%     drawnow
end

S4(5,:) = X_time3_4;
S4(6,:) = Y_time3_4;
```

Listing 12:

```
%%                                      PCA
%%
%                                      Trial 1
    % SVD
A = S1;
mean_val = mean(A,2);
[m,n] = size(A);
A = A - repmat(mean_val, 1, n);
[U,S,V] = svd(A/sqrt(n-1),'econ');

    % Graphs
figure(1);
subplot(2,2,1)
plot(1:n,S1);
set(gca,'Fontsize',13);
title('Trial 1: Original Data');
xlabel('Frame Number');
ylabel('Position (pixel in frame)');

% figure(12)
subplot(2,2,2)
plot(1:n, V(:,1:2))
set(gca,'Fontsize',13);
title('Trial 1: Time Series');
xlabel('Frame Number');
ylabel('Position (pixel in frame)');

% figure(13);
subplot(2,2,3)
plot((diag(S).^2/sum(diag(S).^2)*100), 'ob')
set(gca,'Fontsize',13)
title('Trial 1: Single Values')
xlabel('Index of Single Value')
ylabel('Energy of Single Value')

% figure(14);
subplot(2,2,4)
plot(A'*U(:,1:2))
set(gca,'Fontsize',13)
title('Trial 1: Projected Data')
xlabel('Frame Number')
ylabel('Position (pixel in frame)')
```

Listing 13:

```
%%
%                                   Trial 2

    % SVD
A = S2;
mean_val = mean(A,2);
[m,n] = size(A);
A = A - repmat(mean_val, 1, n);
[U,S,V] = svd(A/sqrt(n-1),'econ');


    % Graphs
figure(2);
subplot(2,2,1)
plot(1:n,A);
set(gca,'Fontsize',13);
title('Trial 2: Original Data');
xlabel('Frame Number');
ylabel('Position (pixel in frame)');

% figure(22)
subplot(2,2,2)
plot(V(:, 1:2));
set(gca,'Fontsize',13);
title('Trial 2: Time Series');
xlabel('Frame Number');
ylabel('Position (pixel in frame)');

% figure(23);
subplot(2,2,3)
plot((diag(S).^2/sum(diag(S).^2)*100), 'ob')
set(gca,'Fontsize',13)
title('Trial 2: Single Values')
xlabel('Index of Single Value')
ylabel('Energy of Single Value')

% figure(24);
subplot(2,2,4)
plot(A'*U(:,1:2))
set(gca,'Fontsize',13)
title('Trial 2: Projected Data')
xlabel('Frame Number')
ylabel('Position (pixel in frame)')
```

Listing 14:

```
%%
%                                   Trial 3

    % SVD
A = S3;
mean_val = mean(A,2);
[m,n] = size(A);
A = A - repmat(mean_val, 1, n);
[U,S,V] = svd(A/sqrt(n-1),'econ');


    % Graphs
figure(3);
subplot(2,2,1)
plot(1:n,A);
set(gca,'Fontsize',13);
title('Trial 3: Original Data');
xlabel('Frame Number');
ylabel('Position (pixel in frame)');

% figure(32)
subplot(2,2,2)
plot(V(:, 1:3));
set(gca,'Fontsize',13);
title('Trial 3: Time Series');
xlabel('Frame Number');
ylabel('Position (pixel in frame)');

% figure(33);
subplot(2,2,3)
plot((diag(S).^2/sum(diag(S).^2)*100), 'ob')
set(gca,'Fontsize',13)
title('Trial 3: Single Values')
xlabel('Index of Single Value')
ylabel('Energy of Single Value')

% figure(34);
subplot(2,2,4)
plot(A'*U(:,1:3))
set(gca,'Fontsize',13)
title('Trial 3: Projected Data')
xlabel('Frame Number')
ylabel('Position (pixel in frame)')
```

Listing 15:

```
%%
%                                   cam3_2                          2

% close all
load('cam3_2.mat')
% implay(vidFrames3_2)

GrayFrameAll = vidFrames3_2(:,:,:,18:18+301);
numFrames = size(GrayFrameAll,4);
X_time3_2 = zeros(1,numFrames);
Y_time3_2 = zeros(1,numFrames);

% initial max
x = 412;
y = 247;

for j = 1:numFrames
    GrayFrame = rgb2gray(GrayFrameAll(:,:,:,j));

    % x filter
    GrayFrame(:,1:x-70) = 0;
    GrayFrame(:,x+30:end) = 0;
    % y filter
    GrayFrame(1:y-50,:) = 0;
    GrayFrame(y+50:end,:) = 0;

    [Max, index] = max(GrayFrame(:));
    [y,x] = ind2sub(size(GrayFrame), index);
    X_time3_2(j) = x;
    Y_time3_2(j) = y;

%     figure(1);
%     imshow(GrayFrame);
%     hold on
%     plot(X_time3_2(j), Y_time3_2(j), 'bo', 'markersize', 10);
%     drawnow
end

S2(5,:) = X_time3_2;
S2(6,:) = Y_time3_2;
```

Listing 16: