Hongyu Cao

Department of Mathematics

University of Washington

**ABSTRACT**

Assume we don’t know anything about spring mass system and analyze the motion of the can captured by 3 cameras at different positions. Some cases have noise in measurement and we need to use PVA to reduce the noise.

**I. Introduction**

We totally have 4 cases and in each case the can’s motion was captured by 3 cameras at different positions and stored by a 4-d matrix (a,b,c,d). d means frames or time and during each frame it consists of 3 layer. Each layer have a value for red, green or blue. Since 4 dimensional is too complicated and we are only interested in the motion of the can so we need to find a way to convert it into 2 dimension.

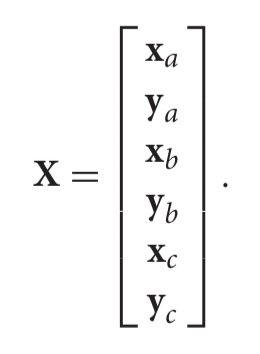
After that because 3 cameras capture the same motion so lots of data are actually redundant so we can use SVD to get a clean data.

II. Theoretical Background

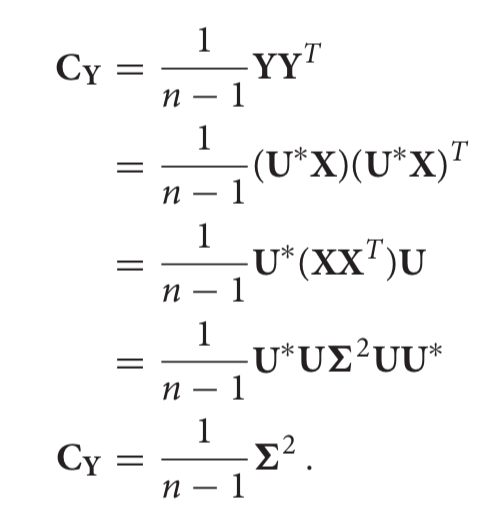
Covariance

The data of the camera can be represented as the following:

camera 1: camera 2: camera 3:

 The reason is that in each camera it has its own plane so we can depict the position of the mass in And when i is different, the plane is also different. Then we can put all of those data into one 1 single matrix:

The data has noise and redundancy and it can be calculated by covariance matrix, which is : Since we want only has large diagonal terms and 0 on off diagonal, using SVD is one way to realize it by changing its basis. Here’s is why: Defining the transformed variable: Y = U\*X and we compute the variance in Y:



**III. Algorithm Implementation and Development**

First of all, since we are only interested in the motion of the can, color does not matter so I converted the original picture into gray scale by the command rgb2gray.

So each pixel only has value from 0(darkest) to 255(brightest), and there’s a flashlight on the top of the can so we only need to try to find the maximal value in each frame. However, some other parts also have value 255. In order to be precise capture only the flashlight, we need to first observe the video and chop it so that the area after chopping only contains flashlight as the brightest object.

I implemented the tracking by a function called track(A,x,y,x1,y1) A is the original matrix and x, y are the cutting lines of rows and x1, y1 are the cutting lines of columns.

Then we need to find the position of the flashlight by finding the area of maximum value. Since there is more than one pixel having the maximum value so we need to get the mean value of the position.

After tracking the moving position of the object, we put all of the data into single matrix and do a SVD to analyze it.

**Sec. IV. Computational Results**

Case 1: This is the ideal case, which means there’s almost no camera shaking and the flashlight was moving up and down. After doing a SVD, I plotted the percentage of each mode:

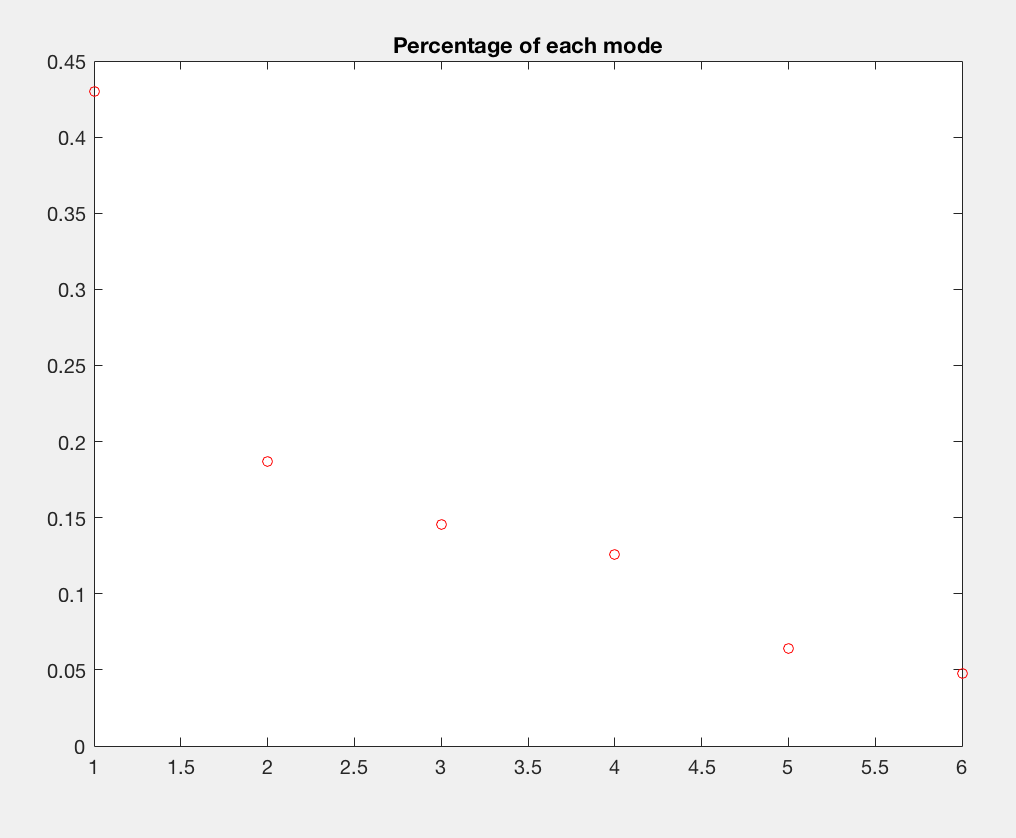


Figure 1: First mode has much more weight than the left ones

And we can just plot the principle component from the mode:

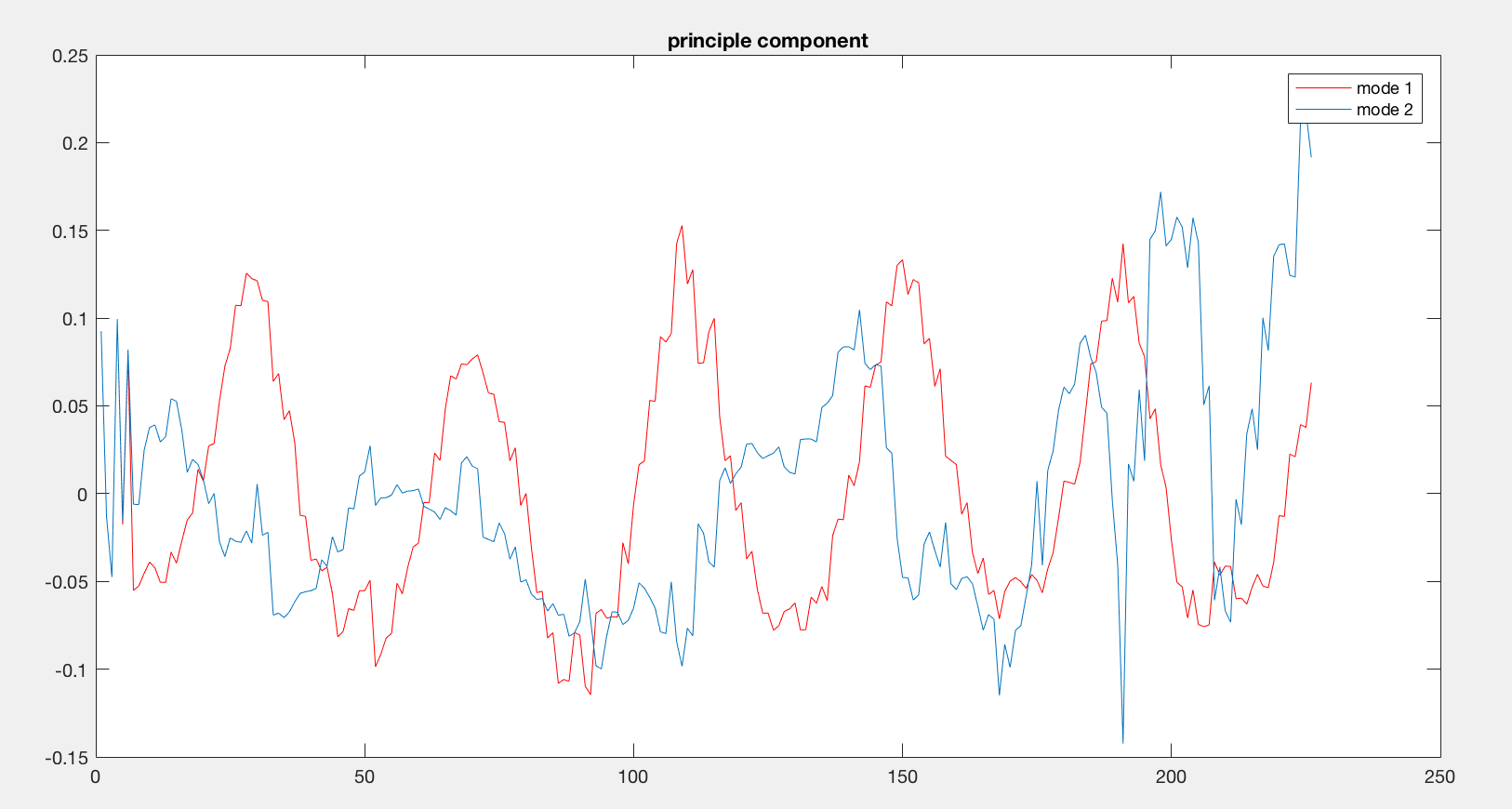


Figure 2: The red line is the first mode and we can see clearly the object is moving up and down. But the mode 2 is useless.

Case 2: This time it has camera shaking, which causes noise of the data.

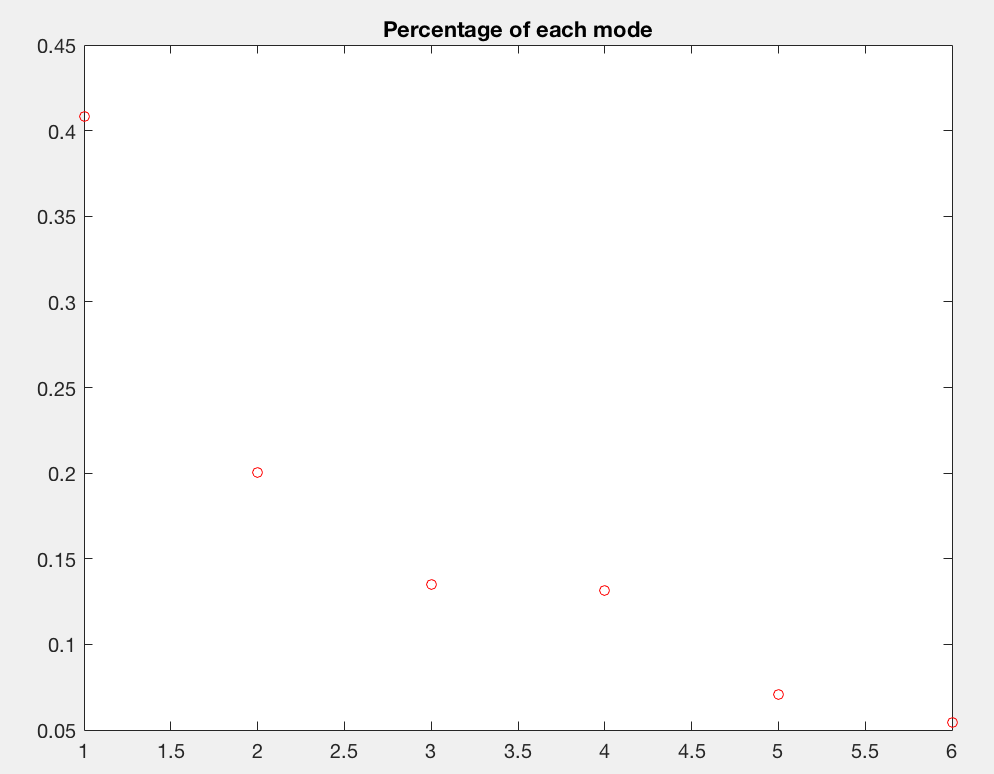


Figure 3: The first mode is still the dominant one

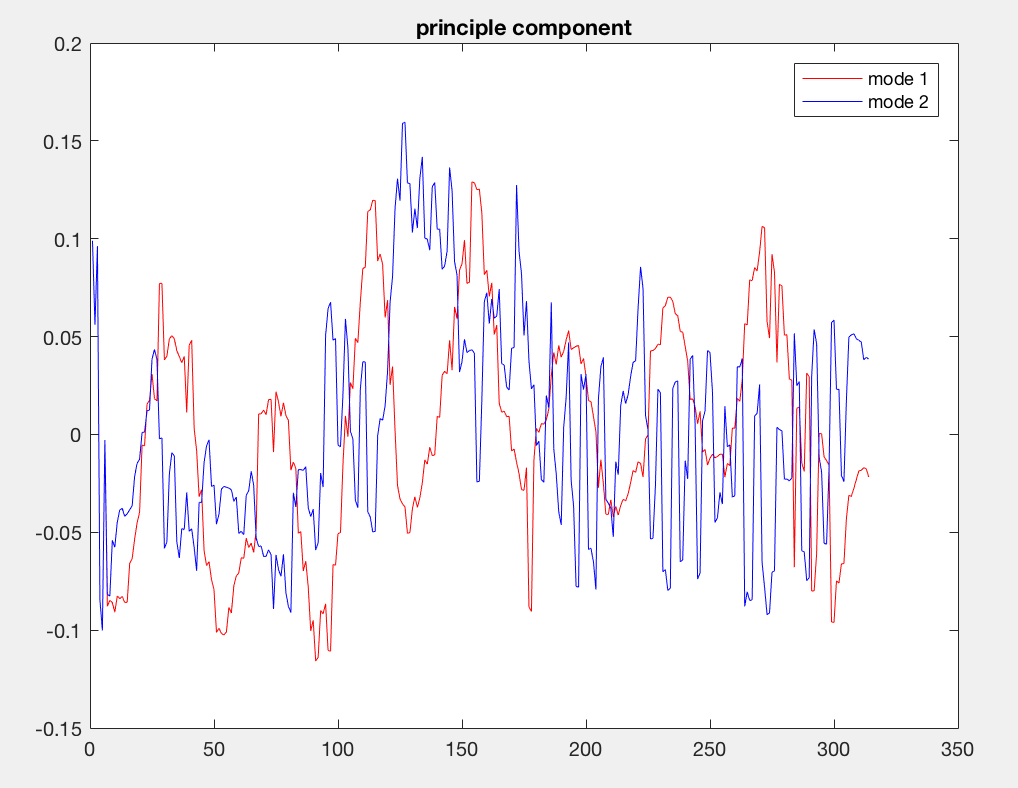


Figure 4: Even this is the case with the noise, we can still see from the mode 1 that the object is moving up and down.

But the first mode is still good enough to depict the motion

Case 3: The mass is released off-center so as to produce motion in the x−y plane as well as the z direction.

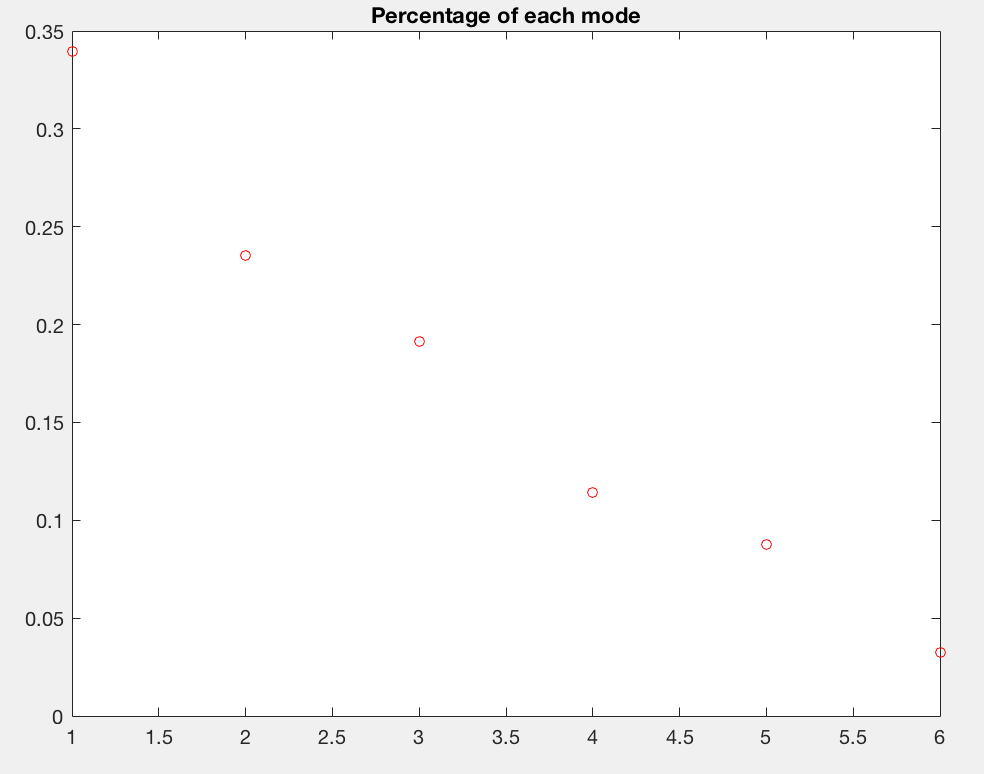


Figure 5: Even the first mode is the biggest, the second is also close to the first one.

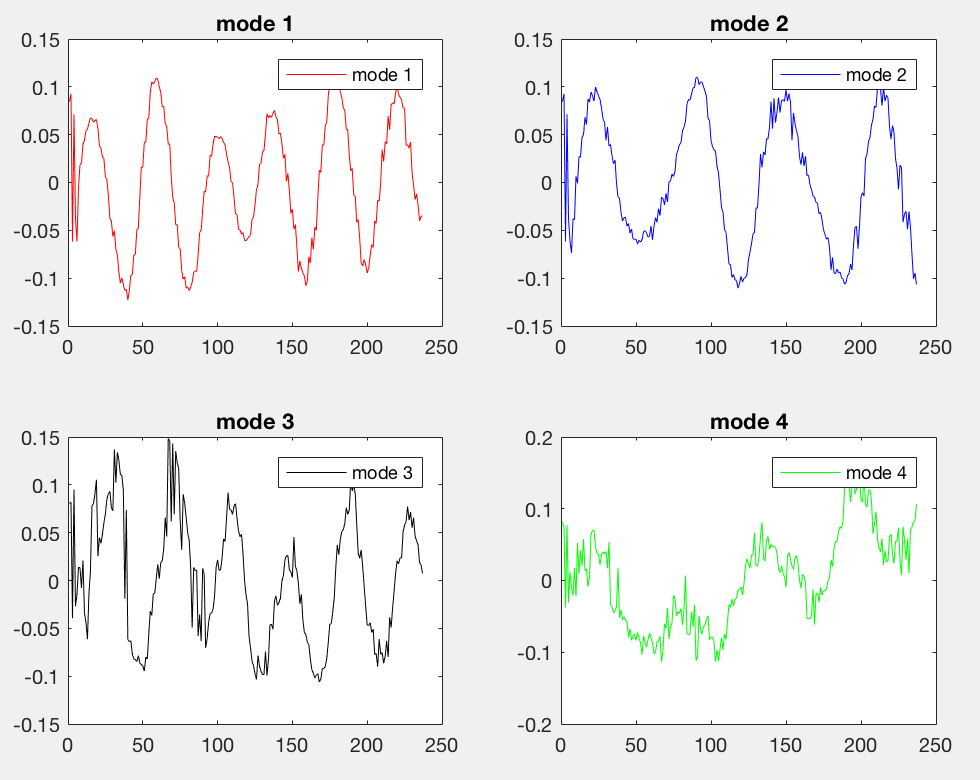


Figure 6: Mode 1,2,3 show the oscillation

The reason of this plot is because the mass is not only moving in one axis so single mode can no longer capture the motion.

Case 4: the mass is released off-center and rotates so as to produce motion in the x−y plane, rotation as well as the z direction.

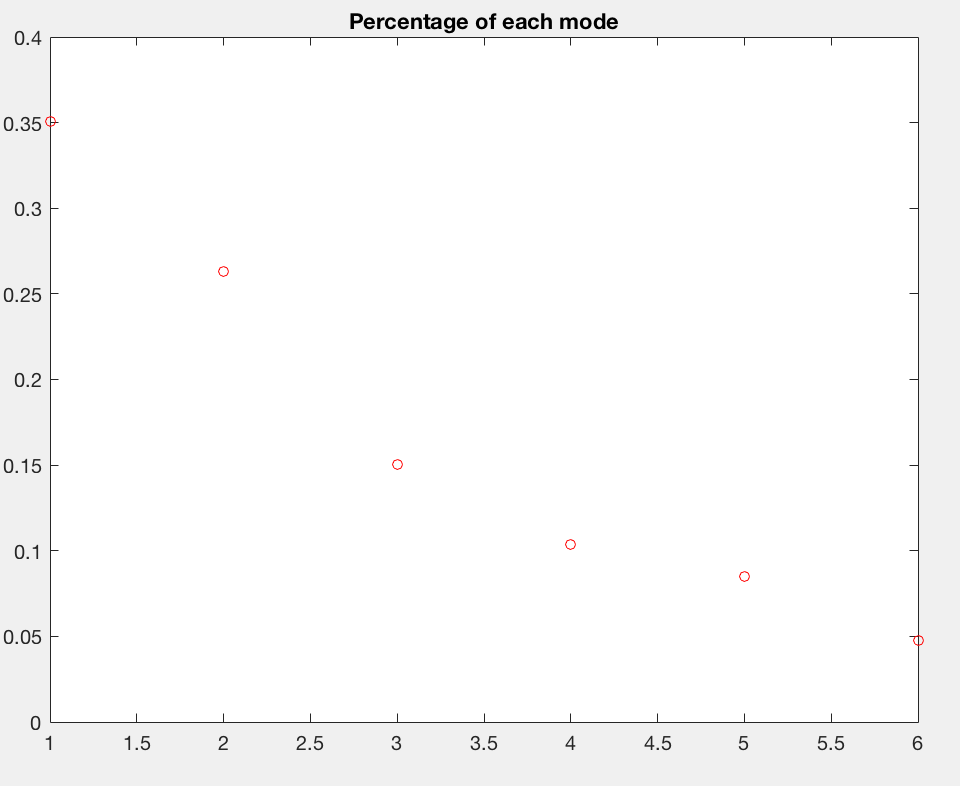


Figure 7: The first mode is only 0.35.

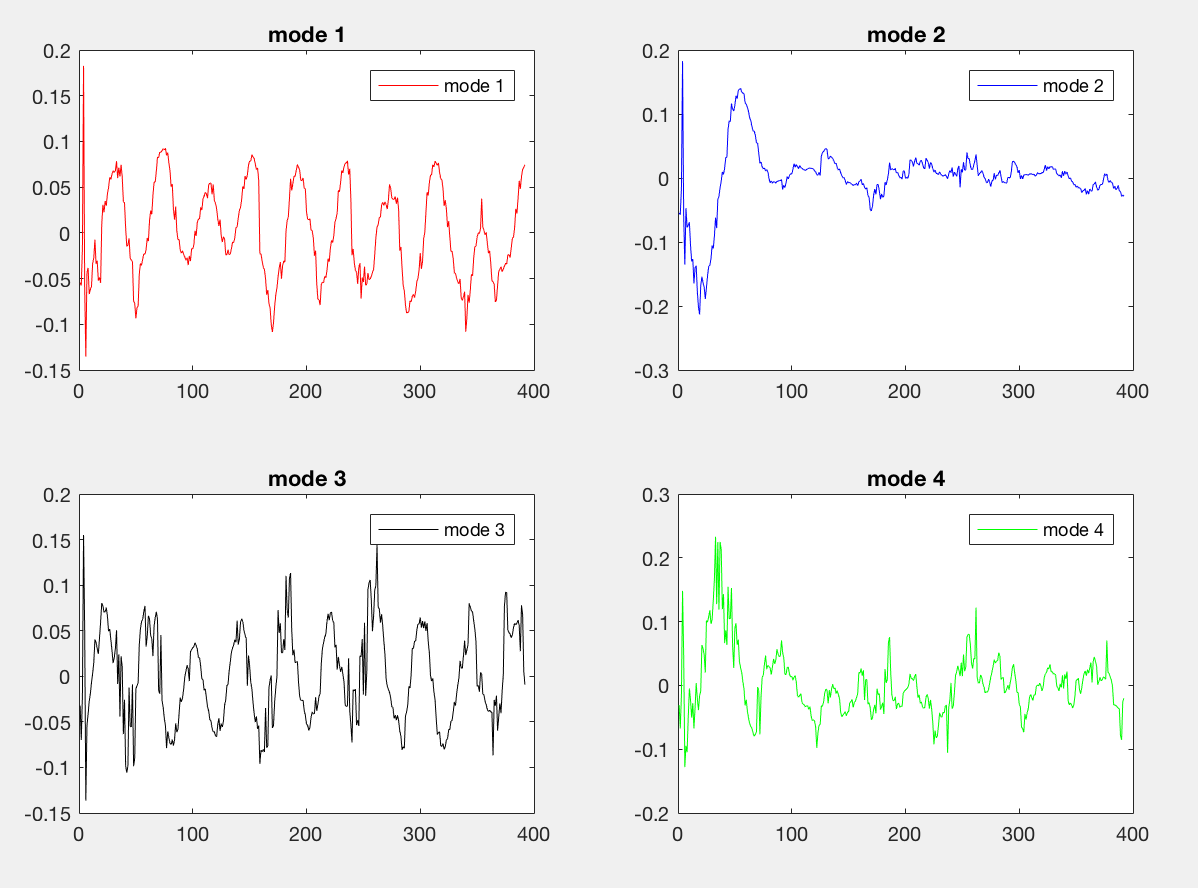


Figure 8: From mode 1 and mode 3, we can see the pendulum motion clearly.

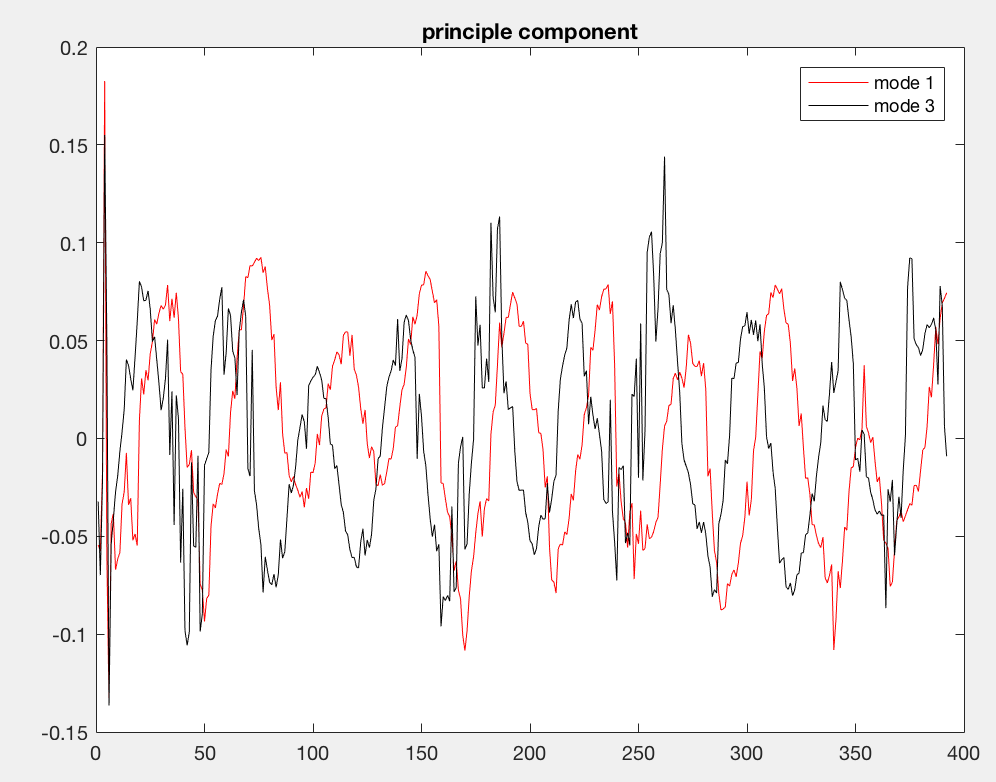


Figure 9: Just compare mode 1 and 3 and we find mode 1 always reaches peak right after the mode 3

From figure 9, mode 1 and 3 indicate 2 kinds of oscillations.

**Sec. V. Summary and Conclusions**

From all cases, we can see that SVD reduce the noise and redundancy of the data and SVD is a efficient way to realize PCA. And first percentage of first mode decrease as the motion becomes more complex since it needs more principle component to express.

**Appendix A MATLAB functions used and brief implementation explanation**

SVD – Singular Value Decomposition into U, S, V

Function tract: It takes parameter A, the original matrix, x, y, the selected area between rows, x1, y1, the selected area between columns and t, the selected time frame from 1 to t

function [D] = track(A,x,y,x1,y1,t)

D=zeros(2,t);

for i = [1:t]

G =rgb2gray(A(:,:,:,i));

C=G(x:y,x1:y1);

[rows,col]=find(C==max(max(C)));

D(1,i)=mean(rows);

D(2,i)=mean(col);

end

end

**Appendix B MATLAB codes**

%% Case 1

A1=vidFrames1\_1;

A2=vidFrames2\_1;

A3=vidFrames3\_1;

t=226;

D1=track(A1,220,410,300,370,t);

D2=track(A2,90,390,253,345,t);

D3=track(A3,235,325,243,487,t);

%% Case 2

A1=vidFrames1\_2;

A2=vidFrames2\_2;

A3=vidFrames3\_2;

t=314;

D1=track(A1,190,400,300,430,t);

D2=track(A2,50,415,190,432,t);

D3=track(A3,190,320,280,460,t);

%% Case 3

A1=vidFrames1\_3;

A2=vidFrames2\_3;

A3=vidFrames3\_3;

t=237;

D1=track(A1,240,420,295,375,t);

D2=track(A2,170,375,220,375,t);

D3=track(A3,190,310,175,485,t);

%% Case 4

A1=vidFrames1\_4;

A2=vidFrames2\_4;

A3=vidFrames3\_4;

t=392;

D1=track(A1,228,438,322,462,t);

D2=track(A2,90,355,231,417,t);

D3=track(A3,166,292,310,476,t);

%% SVD

B=[D1;D2;D3];

m = mean(B,2);

m=repmat(m,1,t);

B = B- m;

[u,s,v]=svd(B);

%%

subplot(2,3,1),plot(B(1,:))

subplot(2,3,2),plot(B(3,:))

subplot(2,3,3),plot(B(5,:))

subplot(2,3,4),plot(B(2,:))

subplot(2,3,5),plot(B(4,:))

subplot(2,3,6),plot(B(6,:))

%%

figure(1)

plot(diag(s)/sum(diag(s)),'ro','Linewidth',[0.5],'MarkerSize', 5), title('Percentage of each mode')

figure(2)

plot (v(1,:),'red'), hold on

plot (v(2,:),'blue')

plot(v(3,:),'black'),legend('mode 1','mode 2','mode 3'),title('principle component')

%% Plot Case 3,4

figure(1)

plot(diag(s)/sum(diag(s)),'ro','Linewidth',[0.5],'MarkerSize', 5), title('Percentage of each mode')

figure(2),

subplot(2,2,1),plot (v(1,:),'red'),legend('mode 1'),title('mode 1')

subplot(2,2,2),plot (v(2,:),'blue'),legend('mode 2'),title('mode 2')

subplot(2,2,3),plot(v(3,:),'black'),legend('mode 3'),title('mode 3')

subplot(2,2,4),plot(v(4,:),'green'),legend('mode 4'),title('mode 4')