MATG 511: Final Report (Revised), on the Topic of Dynamic Mode Decomposition Techniques for Video Footage

Joshua Herman

January 6, 2020

Contents

List of Figures

1 Introduction and Background Information

1.1 Overview of Problem to Solve

This project will be focused on applying techniques from the research paper *Dynamic Mode Decomposition for Real-Time Background/Foreground Separation in Video, by J. Grosek and J. N. Kutz* [1], to perform foreground-background separation on a five minute long surveillance video from the VIRAT video dataset sample release.

Foreground-Background separation is a method that is specifically meant to be used on video footage that has a non-moving background and a moving section that is referred to as the foreground. Footage of this sort, such as that taken by non-moving security video cameras, can be processed through this method to clearly separate the moving parts, such as the people and cars(foreground), from the non-moving parts, such as

sidewalks, buildings, the sky (background). Foreground-Background Separation is necessary in various machine learning project pipelines, as the separation of moving objects from static background may be a good starting point for a project that demands the classification of moving objects; in order to classify individual moving objects, one may find it useful separate those objects from the background. Then other machine learning tools can be used to separate the individual parts from each other and perhaps a sliding windows technique could be used to do so.

1.2 Connection to related work

The Foreground-Background Separation technique implementation is in large part be an attempt at recreation of the results detailed in the research paper by J. Grosek and J. Kutz [1].

2 Methods

2.1 Explanation of the mathematical tools employed

The method to be used for each of the topics addressed in this proposal is Dynamic Mode Decomposition (DMD), whose derivation is given in section 3.1 of the research paper previously mentioned by J. Grosek and J. Kutz [1]. The derivation won't be dealt with here, as the paper gives a pretty intuitive summary.

The problem that DMD is most directly trying to solve is to find a simplified model representation of a timeseries that is decomposable into a set of mode functions, using the tools of linear algebra. Let there be time series, which in this case is a series of m ordered n dimensional vectors x_i , where i=1,2,...,m. The vectors $\vec{x_i}$ i=1...m-1 can be collected into a matrix X and $\vec{x_j}$ j=2...m can be collected into a matrix $\bar{X}$.

To find a model for the data, several steps must be taken. These steps are taken from the November 11 2019 Lecture taught by L. Udeigwe [4]. The list is below as follows:

1. Singular Value Decomposition This is used to convert X into a set of unitary and diagonal matrices:
$$X = U\Sigma V^*$$
where the dimensionality is changed to:
$$U = U_{(n\times l)},$$
$$\Sigma = \Sigma_{(l)\times(l)},$$
$$V = V_{(m-1)\times(l)}.$$
2. We then calculate
$$A = U^*\bar{X}V\Sigma^{-1}.$$

3. Perform eigenvalue decomposition on A, to get W and Λ such that

$$AW = A\Lambda$$

where Λ is diagonal. Note the dimensionality

$$\Lambda = [\Lambda]_{r \times r}.$$

4. We then calculate

$$\Phi = \bar{X} V \Sigma^{-1} W.$$

We then find

$$B = \Phi^{+} \vec{x}_0$$

, where we just introduced the pseudoinverse operator $+$,

$$X^{+} = (X^{*}X)^{-1}X^{*}.$$

Now we have our solution:

$$\vec{x}_i = \Phi \Lambda^{i-1} B$$

. Our Matrices can be written in terms of individual components:

$$\Phi = [\vec{\phi}_1, ..., \vec{\phi}_r].$$

$$B = [b_1, ..., b_r]^T.$$

, and

$$\Lambda = diag([\lambda_1, ..., \lambda_r]).$$

. Rewriting our solution in component form, we have:

$$\vec{x}_i = \sum_{j=1}^{r} \phi_j \lambda_j^{i-1} b_j.$$

Our frequencies are

$$\omega_j = log(\lambda_j)/\Delta t$$

, where Δt is the time between consecutive vectors $\vec{x}_i$.

2.2 Application to Project

In this project, the video data will be converted into several X_{full} matrices where each column contains the image information of a separate frame. The frames are assumed to be equally spaced in time. The DMD method thus takes

$$X_{full} = [\vec{x}_1, ..., \vec{x}_m]$$

as its input and returns Φ,Λ, and B, where approximately

$$\vec{x}_i = \Phi \Lambda^{i-1} B.$$

The time multipliers are

$$\Lambda = diag([\lambda_1, ..., \lambda_r])$$

. Ourfrequencies are

$$\omega_j = log(\lambda_j)/\Delta t$$

. Note that these are all complex valued. Our Solution in component form

$$\vec{x}_i = \sum_{j=1}^{r} \phi_j \lambda_j^{i-1} b_j,$$

is a dynamic mode decomposition. The j^{th} mode is

$$\phi_j \lambda_j^{i-1} b_j = \phi_j exp[\omega_j t] b_j.$$

It is very evident that λ_j (or ω_j) is the term from the individual mode that is solely responsible for the j^{th} mode's dynamic behavior. Thus it can be called the frequency term. The vector ϕ_j is the only vector term in the mode. This can be though as the shape of the mode, and it determines the static behavior of its associated mode. The coefficient term b_j is just a weighting factor of sorts, that takes into account the initial condition.

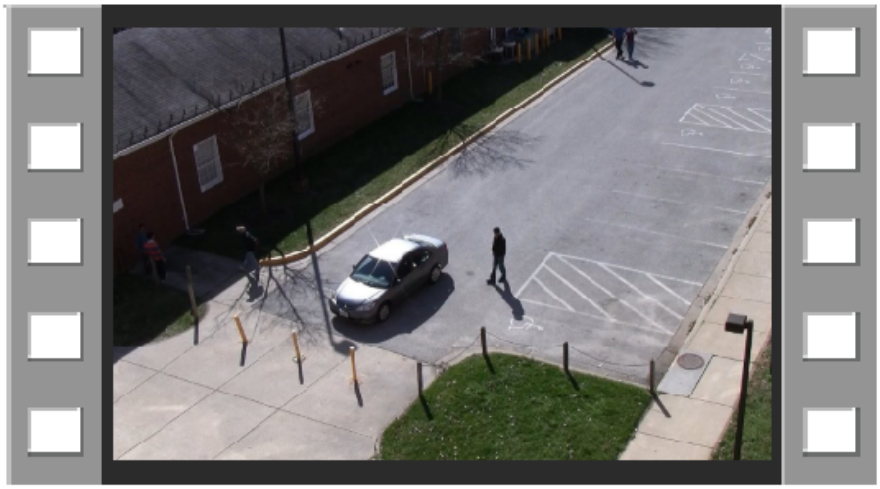
Each mode can be thought of as a separate video or picture that evolves with time, where the sum of all modes at an instant yields the original picture at that instant. The mode that evolves the least in time is the mode that represents the most stationary video from the set. This mode is the mode whose continuous frequency magnitude $||\omega_j||$ is smallest. The rest of the modes are more representative of the more dynamic parts of the video. Sorting the modes by order of increased frequency magnitude will done in the analysis. The mode with index k that has the smallest $||\omega_k||$ will be dubbed as the background due to its being the least dynamic, while the rest of the modes can be considered part of the foreground due to their being more dynamic. This reasoning was explained in Section 3.2 P2 of the paper by J.Grosek. [1].Thus the foreground modes will be added together to produce the foreground images, while the background mode will be used to produce the background image. Note that only the real component of the pixel values will be considered. Also, the 256 mod of the sum of the real values will be taken, to make sure all pixels are confined to a readable pixel range[1].

2.3 Description of Dataset

The dataset to be considered is a several minute long surveillance video file from the VIRAT Video Dataset, which is described on its associated website as being a benchmark dataset for surveillance video tasks in computer vision.[3] The Sample of Release 1 of the dataset will be used[2]. The file is called, *VIRAT_S_000002_sizesmall.mp4* and it is a 270MB file whose video has a time duration of 5 minutes and 2 seconds, a frame rate of 29.97 frames per second. Its resolution is 1920 by 1080, it has 3 color channels, and has a total of 9075 frames.

This particular surveillance video shows footage of a camera fixed on an outdoor parking lot to the side of a building. The background is completely still, but a few people and objects are occasionally moving around. These few people and objects that move around are referred to as the foreground. At the beginning of the footage, several people slowly walk toward the side of a building toward the back of the parking lot. Then a little later a car drives up and parks in the back. Other people then enter the scene and one of them is rolling a wheel barrel as they walk toward the car. Then the trunk opens and an object is put into the wheel barrel. Then they close the trunk and as the car drives away most of the people disperse and the guy with the wheel barrel walks off screen. Several guys still wait near the back of the parking lot.

Figure 1: Surveilance Video VIRAT



2.4 Preparation of your data set

There are three main steps used to preprocess the surveillance video file into sets of matrices that are DMD ready.

1. Convert to downgraded Images
2. Convert Images to Single Matrix

3. Down Sample the number of matrix columns and collect them into subsets of 30 column matrices.

2.4.1 Convert to images

Python video and photo processing packages will be used to complete this step. This step entails having the mp4 file's frames decolorized, and resolution reduced and saved with sequentially numbered file names.

1. The library *pims* is used to conveniently load individual frames through numpy like indexing. Each frame will then be converted to an numpy array. The dimensions of each image array extracted are (1080,1920,3) or (H,W,C) where H is height, W is width, and C is color.

2. The package cv2 will be used to process the arrays to reduce dimensionality, which will greatly simplify DMD analysis. Downscaling the image to grayscale decreasing the resolution by a factor of 10 yields a new picture with a lower dimensionality of (108,192). The process of downscaling is rationalized in the paper by J.Grosek and J.N.Kutz [1] states in Section 4.1 P1 that downsizing the original images and converting them to grayscale was done in their DMD data processing of select videos from the Advanced Video and Signal based Surveillance, was done for the purpose of making the "computational memory requirements manageable for personal computers." Then cv2 is used to save the image.

Figure 2: Extracted Photo Frame 0

Figure 3: Extracted Photo Frame 6000

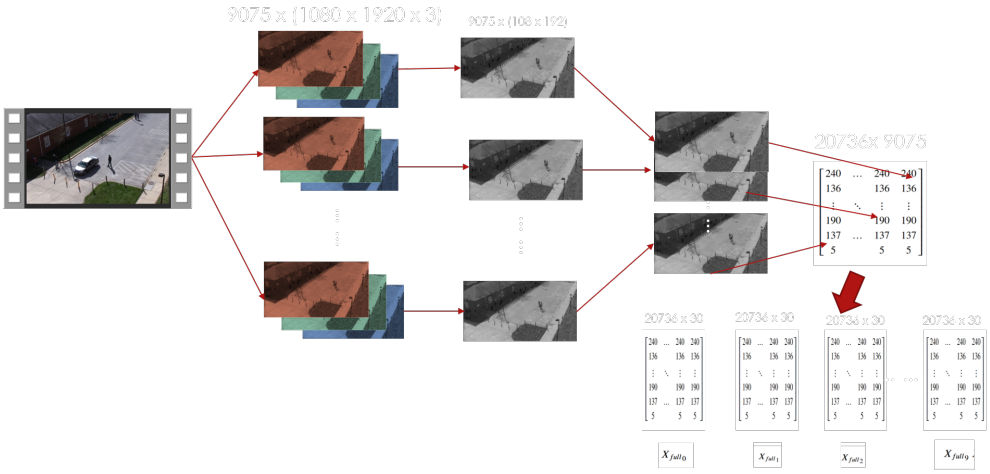
2.4.2 Convert images to numpy matrix

The Skimage.io package will be used to collect all of the images, and concatenate them into a 2D array that can be saved as a csv file, where each column is a different vectorized image. The (Number of Pixels, Number of frames).

2.4.3 Frame Selection and Grouping

The goal here is to create arrays that are primed and ready to be immediately inserted into the DMD algorithm. Section 4.1 P2 of the paper by J. Grosek and J.N Kutz [1] states that their preprocessing involved splitting the video stream into sets of 30. However, in my own analysis it turned out that the frame rate also had to be reduced, in order for the time interval that is spanned by the set of 30 frames to not be too small to contain noticeable motion. Since the frame rate is 29.97 frames per second, down sampling the number of frames by a factor of 30 would be essential, in order for a full 30 seconds worth of motion to be captured in each set of 30 frames. Thus a new array is created containing only every 30th vectorized image from from the saved array. Then this new array will be split into a sets of 30 vectorized images, which will be referred to as the set of arrays X_{full_1}, X_{full_2}, etc. Each of these arrays are now ready to directly entered into the DMD algorithm, whose results are directly in the form of the three arrays: Φ, Λ, B.

Figure 4: Data Preparation Visual



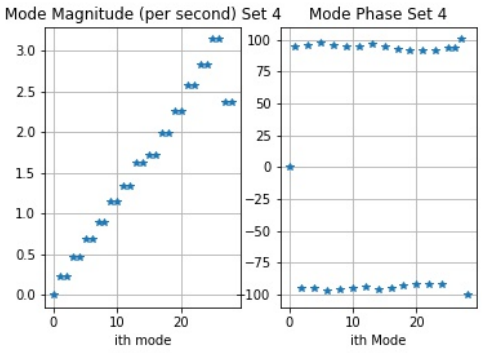
3 Results

There were 9075 images originally extracted. Then every 30th frame was picked out. The resulting 1/30th of them were split into 10 sets of 30 images. Each of the 10 sets were fed into the DMD algorithm. The plots of frequency and extracted images can be seen below.

3.1 Plots for Magnitude and Phase of Frequency ω

For each set, there is a plot below that shows the frequencies for each mode, scatter plotted in terms of magnitude and complex phase.

Figure 5: Plot of Set 4
Magnitude and Phase of Frequency



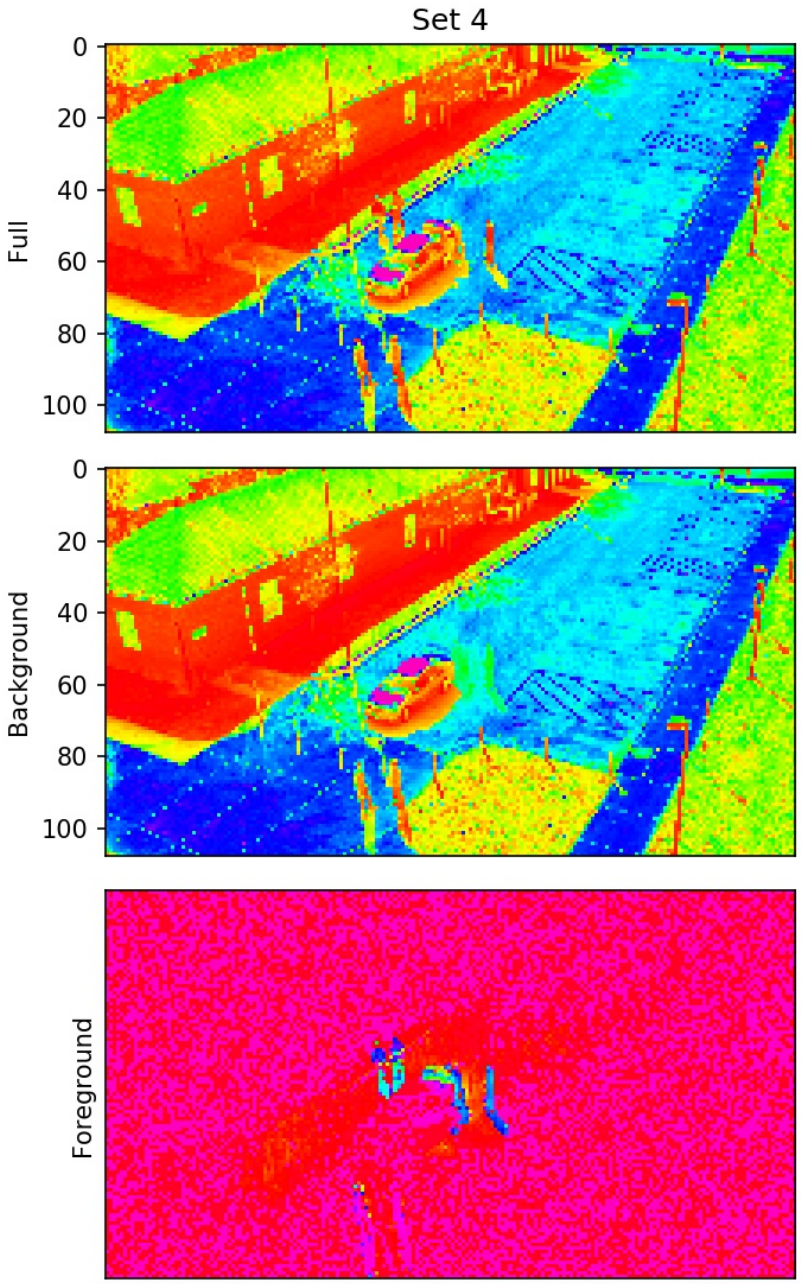
These Plots provide a good benchmark for choosing which mode is foreground vs background. Since the real part of the frequency is responsibble for exponential growth

and decay, while the imaginary part is repsonsible for the oscilatory motion, one quick metric is to take the absolute value of the frequency so that both forms of dynamic behavior can be evaluated at once. Thus, following the above plot on the left, we see that mode 0 has the smallest magnitude of frequency which is zero. This implies no exponential nor oscilatory behavior. Thus we have a good condidate for the background mode. In the process used for automating the background foreground separation in this project, the absolute value (magnitude) of the frequency ω_j is this benchmark. We automaticaly pick the mode with smallest magnigude as our background mode. The rest of them are summed to attain a model of the foreground, which is supposed to have dynamic behavior. The phase of ω is mostly irrelevent in this project, since it doesn't do more than give a glimpse into how much of the frequency is imaginary and how much is real. Based on the phase plot, it would seem that all but the first mode have primarily imaginary frequency, since the phase is close to +-90. This implies that these modes have primarily oscilatory behavior as opposed to exponential behavior, though that may still be misleading as to how insignificant the exponential decay aspect is.

3.2 Plots for Full recomposed image, Background, and Foreground

For each set, there is a stacked plot below that shows different versions of the set's middle recomposed image. The top version is the full recomposed one, the middle one is the background, and the bottom one is the foreground.

Figure 6: Plot of Set 4
Full,Background,Foreground Images



Above we can see three reconstructed images for the 16th frame of set 4, the first being the fully recomposed one that includes all modes, the second being only the background mode, and the third being the sum of the foreground modes. As se can see, The Foreground contains several people and part of the trunk of a car. The background contains a car, but with a small piece missing and no peole standing directly around it. The full image contains all of these people and all of the car. Thus there is sucess in this particular implimentation.

Below is the background picture for set 4. Also, below is the foreground for frames 156 to 172 (out of 300), which correspond to part of set 4. You can see how the car gradualy drives away and shrinks out of view.

Figure 7: Background Picture 152

Figure 8: Foreground Picture 156

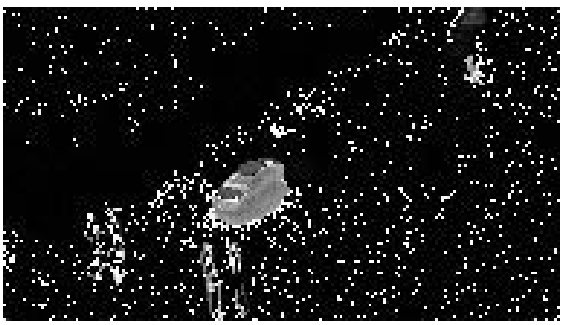


Figure 9: Foreground Picture 157

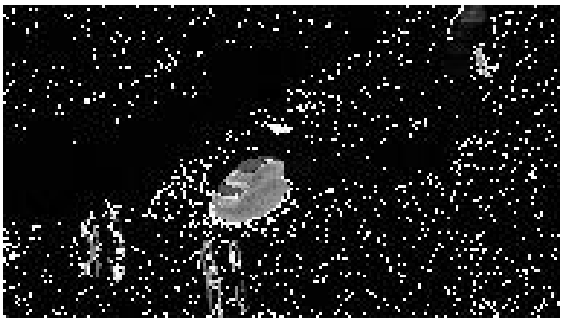


Figure 10: Foreground Picture 158

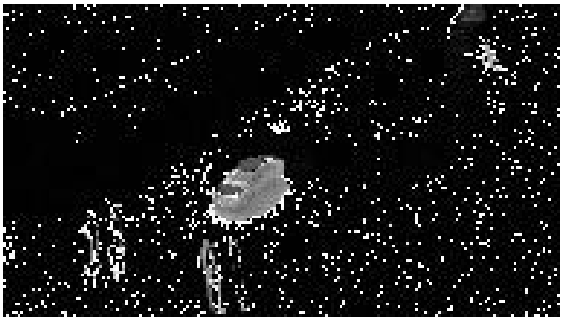


Figure 11: Foreground Picture 159

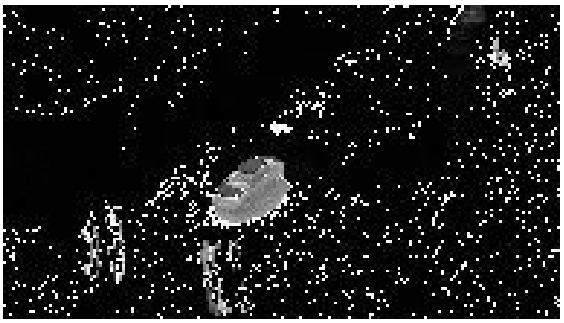


Figure 12: Foreground Picture 160

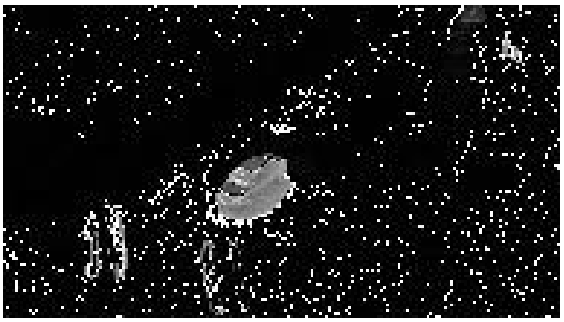


Figure 13: Foreground Picture 161

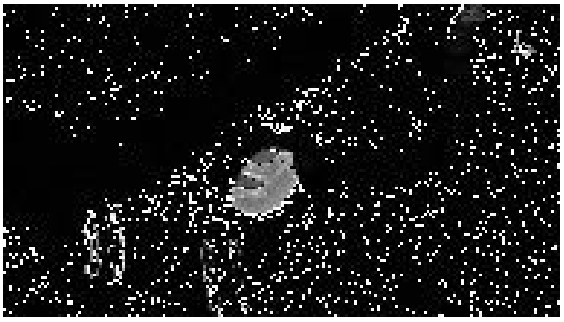


Figure 14: Foreground Picture 162

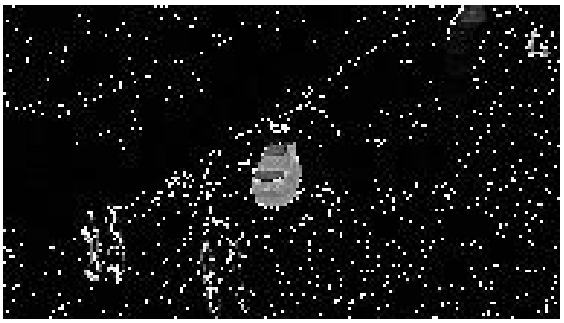


Figure 15: Foreground Picture 163

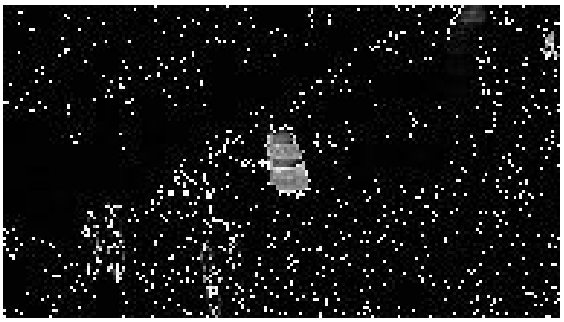


Figure 16: Foreground Picture 164

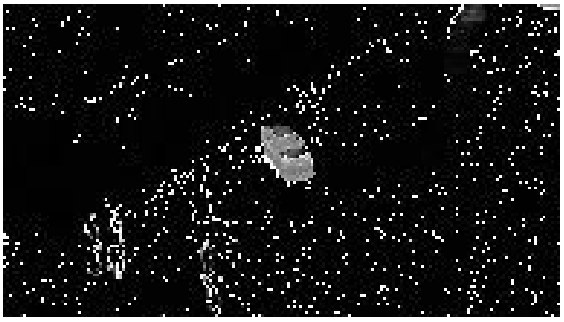


Figure 17: Foreground Picture 165

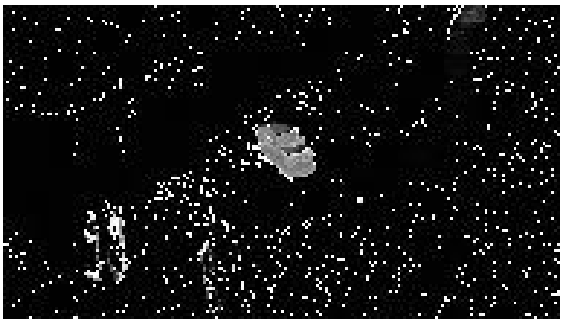


Figure 18: Foreground Picture 166

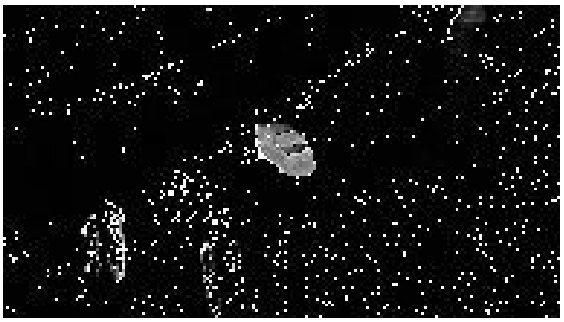


Figure 19: Foreground Picture 167

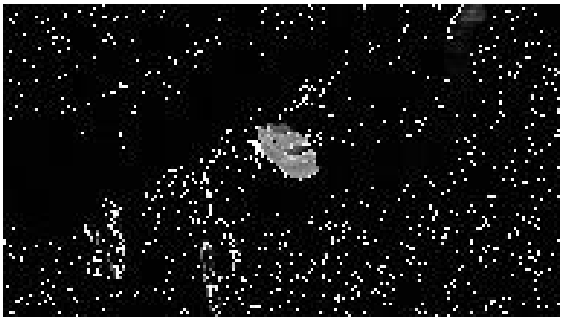


Figure 20: Foreground Picture 168

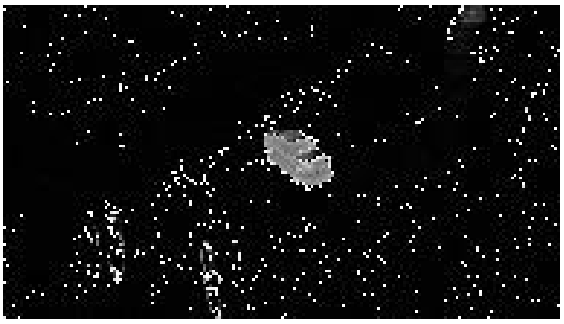


Figure 21: Foreground Picture 169

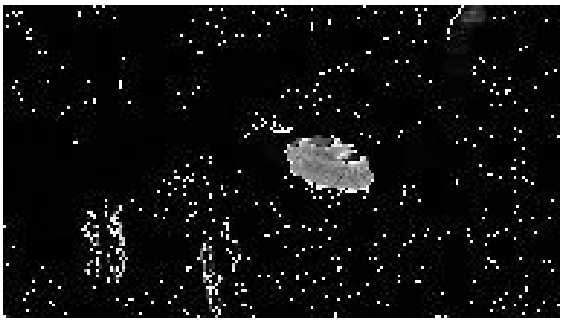


Figure 22: Foreground Picture 170

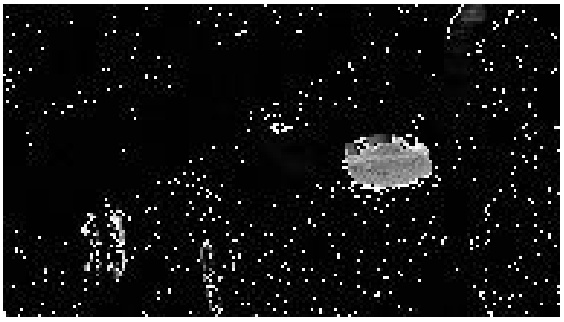


Figure 23: Foreground Picture 171

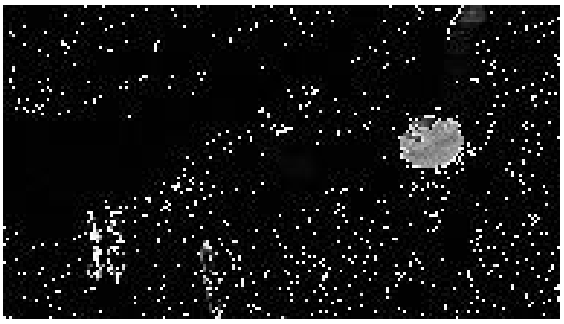
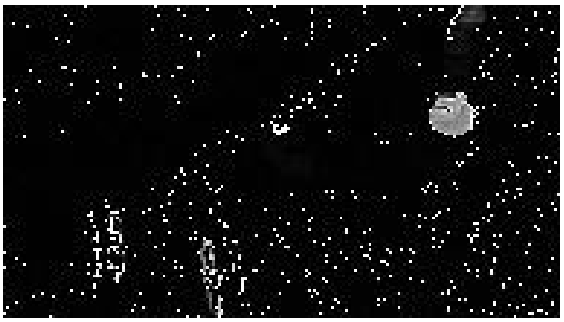


Figure 24: Foreground Picture 172



4 Analysis of Redone Results

This time around the process of datapreparation and DMD are redone, but only considering the top right corner of the image near the parking lot entrance

Figure 25: Split video into 9 sections. Pick top right, outlined in red.

We perform all of the same steps as before, but downsample to include every 15 frames, to reduce the frame rate to 2FPS. We will mainly consider analyzing individual modes using complex analysis that result from Dynamic Mode Decomposition being performed only on that one section.

First, some essential math will be briefly discussed. The kth mode is

$$\vec{\phi}_k {\lambda_k}^{n-1} b_k.$$

This mode can be broken down into the product of an amplitude, exponential, and oscilatory term using complex analysis. Noting how for any complex vector α,

$$\alpha = |\alpha| e^{i * Phase(\alpha)} = |\alpha| cis(Phase(\alpha)).$$

Below is a mathematical manipulation of the mode to into magnitude times phase format.

$$\vec{\phi}_k {\lambda_k}^{n-1} b_k = |b_k \vec{\phi}_k| |\lambda_k|^{n-1} exp(i * [Phase(b_k \vec{\phi}_k) + (n-1) Phase(\lambda_k)])$$

$$= |b_k\vec{\phi}_k||\lambda_k|^{n-1}cis([Phase(b_k\vec{\phi}_k) + (n-1)Phase(\lambda_k)])$$

The real component is the only component of the pixel values that we consider when recomposing the image. Below is the real component of the mode.

$$\Re(\vec{\phi}_k{\lambda_k}^{n-1}b_k) = |b_k\vec{\phi}_k||\lambda_k|^{n-1}cos([Phase(b_k\vec{\phi}_k) + (n-1)Phase(\lambda_k)]).$$

We can see that $b_k\vec{\phi}_k|\lambda_k|^{n-1}$ is the amplitude of the cosine wave at frame n, while $[Phase(b_k\vec{\phi}_k) + (n-1)Phase(\lambda_k)]$ is the angle of that wave at frame n.

We can see that the equation for the value of the vector of pixels in mode k is an exponentialy decaying cosine wave.

1. The term $|\lambda_k|^{n-1}$ causes the wave amplitude to be multiplied by a factor of $|\lambda_k|$ when n increases by 1, assuming $|\lambda_k| < 1$. Thus this paper will refer to $|\lambda_k|$ as the Pixel time Multiplier.

2. The term $|b_k\vec{\phi}_k|$ is the amplitude when $n = 1$. Thus we will call it the pixel amplitude.

3. The term $(n-1)Phase(\lambda_k)$ is the term that produces oscilations by increasing the wave angle by $Phase(\lambda_k)$ when n increases by 1. Thus we will refer to it the pixel complex frequency. This term is in radians.

4. The term $[Phase(b_k\vec{\phi}_k)]$ is the term that gives the cosine wave its initial angle when $n = 1$. The angle will be expressed in radians.

Figure 26: The four terms that characterize an individual mode

Now we will discuss a visualiation of the breakdown of Mode 0 from Set 8 into these four parts. Below are figures 27 and 28, which will give a visualization of the first 2 items (Amplitudes) from list 26 mentioned above:

Figure 27: Mode 0 Set 8 Magnitude Heat Map

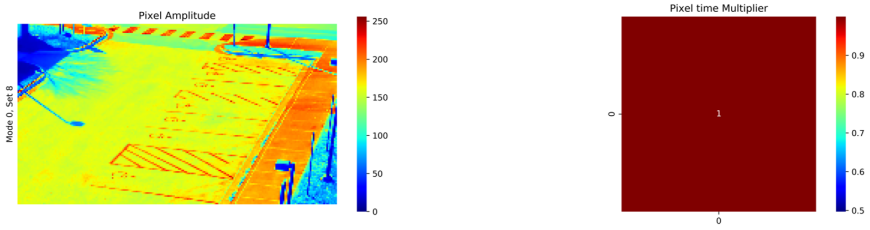


Figure 27, contains two plots. On the left is the pixel amplitude plot corresponding to the term in 26 term 2. This term is a vector, and it is called the pixel amplitude. which when reshaped to the dimensions of a picture and plotted on a heatmap, yields a picture of the parking lot.This picture just from inspection looks like a good candidate for the background mode. This particular colloring was used for ensuring visible color contrast. The bar on the right points out what the number values are for List item 2. We can see that the values run from 0 to around 255. To the right of the plot is a heatmap of the pixel time multiplier , which shows the pixel time multiplier to be 1. From these two pieces of information, it must be that Mode 0 of set 8 has a constant amplitude for each pixel.

Figure 28: Mode 0 Set 8 Phase Heat Map

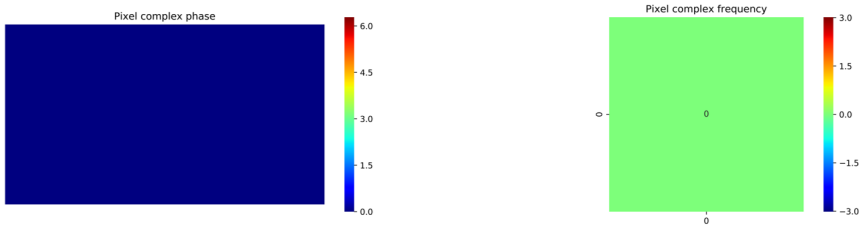
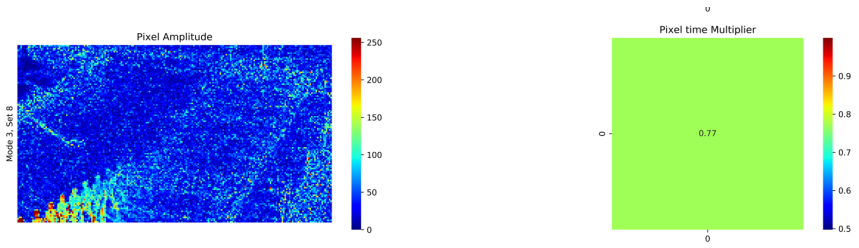


Figure 28 contains a plot on the left, which is list item 44, of each pixelś starting angle, or phase. Figure 27 contains a plot on the right, which is of the pixelś frequency3, which will increase the pixelś angle by the frequency each time n increases by 1. As can be seen, the phase and frequency are both zero, meaning that this mode doesn't oscilate at all the cosine part of the mode equals 1. This proves that The equation for the pixels will thus only be $|b_k\vec{\phi}_k|$.

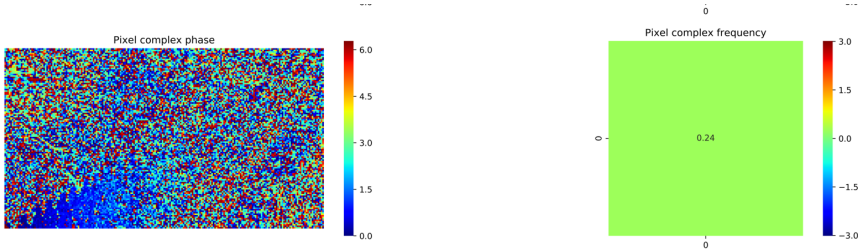
Below are figures 29 and 30, which show the same kind of information for Mode 3 of set 8.

Figure 29: Mode 3 Set 8 Magnitude Heat Map



The plot on the left of figure **??** shows the initial amplitudes, which as you can see shows sihloutes of people on it, with pixel values in the 200s when looking at the bottom left corner. When moving upward diagonaly from that corner, the silhoutes move to lower pixel values. The Right Plot shows that the pixel time multiplier is 0.77, thus meaning that the pixel values are multiplied by 0.77 each time n increases by 1.Doing some simple math $.77^18 = 0.009$ Thus by the 19th out of 30th frame, the picture will become close to irrelevent. This of course doesn't take into account the cosine function, which will be discussed next.

Figure 30: Mode 3 Set 8 Phase Heat Map

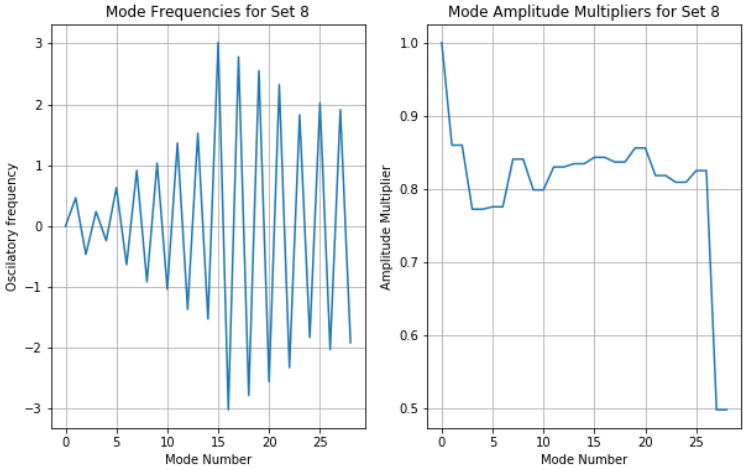


In plot 30, we can can see on the left the starting angle for each pixel, and we can see at the bottom left there are silhoetes that have a phase of near 0. Thus it seems apparant that this specific mode may at in its first frame be a significant contributer to the foreground. The frequency is 0.24, which causes an increase in angel by 0.24 each time n increases by 1. Moving forward 7 frames will increase the phase by around 1.68, which is around $\pi/2$. , which is 1 quarter of a cycle. Within this cycle, all angles will reach a point where cosine be near zero. Thus the pixels will turn off and on depending on what their starting phase is. We can infer that the dark blue silhoutes in the phase heatmap will become close to zero in pixel value by frame 8, and by frame 15 when cosine becomes close to -1, we will also have an original amplitude multiplied by $(.77)^{14} = 0.03$.

Thus the pixels will have negative values that are close to negligible. Also, the mod 256 function will be used to turn these pixels into positive values, which will actualy cause the negligible negative value to be treated as a large positive value, although the mod operation is only considered after all of the modes are summed. In future attempts may want to consider an alternative method of dealing with out of range values.

Now we will show the frequency mode diagram for all modes in Set 8

Figure 31: FreqPlot-Set8



In figure 31, we on the left have the mode frequencies, or $Phase(\lambda_k)$, while on the right we have the mode amplitude multipliers $|\lambda_k|$, plotted for the kth mode. The plot on the left shows that for $k = 0$, the frequency is zero, while the plot on the right shows that for $k = 0$, the amplitude mutiplier is 1. Thus this mode is unchanging, and is consistant with our previous inference of mode 0 being the background mode. However, all other modes have a nonzero frequency and a nonunity amplitude multiplier. So all other modes are dynamic, and thus are considered foreground.

To conclude, the frequency gives an idea of how fast the pixels oscilate, the amplitude multiplier gives us a clue of the exponential decay rate.

4.1 Additional Complex Analysis Concerns.

One small detour will be discussing why breaking the frequency ω into magnitude and phase has limited usefulness in analysis.

$$\lambda_j = exp[\omega_j * \Delta t]$$
$$= |\lambda_j| * cis(Phase(\lambda_j)) = exp(|\omega_j|\Delta t cis(Phase(\omega_j)))$$
$$= exp[\Delta t * |\omega_j| cos(Phase(\omega_j))] * cis(sin(Phase(\omega_j))$$

You can see that the equation is verry unwieldy; we have λ in a form such that it contains its phase twice inside of two double functions. There is no way to get $|\lambda_j|$ back from these terms. Thus, make sure to not overintepret values for the phase and magnitude of ω. However, considering that if the $Phase(\omega_j)$ is zero, then la $\lambda_j = exp[\Delta t * |\omega_j|]$. If $|\omega_j| = 0$, then $\lambda_j = cis(sin(Phase(\omega_j))$ We can see that these relations don't give too much information, and that the magnitude of omega gives the most useful information.

References

[1] J. Grosek and J. N. Kutz Dynamic Mode Decomposition for Real-Time Background/Foreground Separation in Video. arXiv: 1404.7592vl [cs.SV] 30 Apr 2014

[2] $http://www.umiacs.umd.edu/workshops/cvpr/viratdata/VIRAT_Video_Dataset_Release1.0_sample.zip$

[3] $http://viratdata.org/$

[4] L. Udeigwe, MATG 511 Computational Methods Lecture,November 11 2019,DMD algorithm