CHEMICAL PLUME DETECTION FOR HYPERSPECTRAL IMAGING

ABSTRACT. This paper details various aspects of the detection and identification of chemical plumes in long wave infrared (LWIR) data. The lack of well defined edges and the dynamic nature of a gas cloud leads to challenges in detection, particularly when the cloud diffuses and becomes thin. Contemporary graph segmentation algorithms are investigated to track the movement of the gaseous cloud as it spreads through the surround environment. Semi-supervised hyperspectral unmixing is explored as an alternative to probabilistic detectors for the identification of particular chemical signatures. Also, false color representations are explored as a method of visualizing high dimensional LWIR data.

1. INTRODUCTION

The detection of chemical plumes in the atmosphere is a problem that has significant applications to defense, security and environmental protection. The accurate identification and tracking of airborne toxins is crucial to combat the use of chemical gases as weapons, prevent fatalities due to accidental leakage of toxic gases and avoid contamination of the atmosphere. Identification of harmful gases with high fidelity is needed to provide warnings in threatening situations. In these grave scenarios it is crucial to correctly pinpoint the source of these fumes and track the diffusion of dangerous plumes into the atmosphere. Laboratory measured signatures of dangerous chemicals are available to assist in chemical plume identification. However, testing and training data is not readily available due to the inherent danger of these real world situations. Instead, open air testing with surrogate chemicals is conducted to study the diffusion of chemical plumes. The developed plume detection methods must meet strict requirements to ensure the fidelity of a detector.

The hyperspectral data set in this report is used to develop image processing algorithms for chemical plume detection. The objective is to implement different plume detection and segmentation algorithms to analyze the hyperspectral video sequences to find a reliable and robust method that identifies the gaseous plume emissions. To qualify as a viable technique, it is desirable for a detector to be accurate and computationally efficient. Various image processing algorithms have been tested to attempt to visualize the long wave infrared data, and subsequently unmix the dominant signatures in the scene and also separate the chemical plume from the terrain in the background.

2. Overview

The data we are working with consists of calibrated hyperspectral data cubes collected during 2006 from the Dugway Proving Ground in Utah. Each dataset is a video sequence with between 100-300 frames.

Since it is not in the interest of public health to release toxic chemicals into the atmosphere, the samples that were released were non-toxic and chosen to have emissivity signatures close to those of known toxic chemicals. They were taken with an Fourier Transform Infrared based long wave infrared sensor.

The hyperspectral data set analyzed for this project was provided by the Applied Physics Laboratory at Johns Hopkins University. The hyperspectral data consists of a series of video sequences recording the release of chemical plumes into the atmosphere. Figure 1 shows the three long wave infrared spectrometers (named Romeo, Victory and Tango) placed at different locations to track the release of known chemicals. The frame rate of these sensors is 0.2 Hz and for each instance in time, the long wave infrared sensor took three dimensional hyperspectral data. Each of these data cubes has two spatial dimensions of 128 $\times$ 320 pixels, while the third dimension is the spectral dimension, measuring the spectral radiance at 129 wavelength channels in the electromagnetic spectrum. Each layer depicts a particular frequency of the long wave infrared starting at 7,830 nm and ending with 11,700 nm.

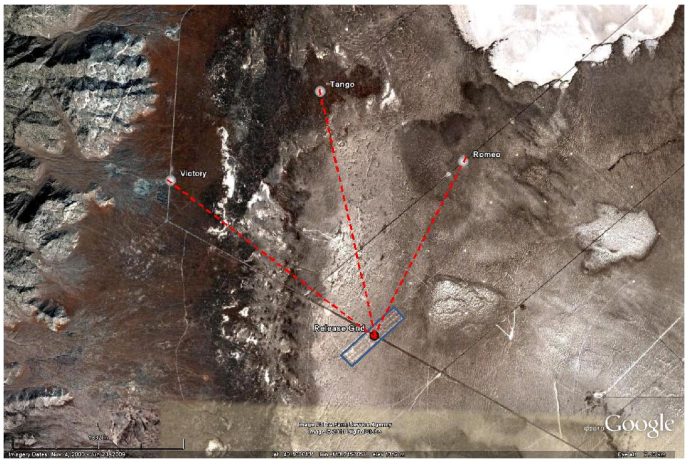


Figure 1. Placement of the three long wave infrared spectrometers less than 3 kilometers from the plume release.

The ground-based long wave infrared sensors measure the spectral radiance of the scene. Radiance measures the amount of radiation emitted from a surface, and is technically defined to be the electromagnetic flux per unit projected area per unit solid angle. In particular, the spectral radiance measures radiance at particular wavenumbers in the electromagnetic spectrum

in units of $\mathrm{Wm}^{-2}\mathrm{sr}^{-1}\mathrm{cm}$. The long wave infrared data set is not visible to the human eye without further image processing. The high dimensionality and type of data in this hyperspectral data set posed significant challenges in computation time of each algorithm. Without the capability to directly visualize the chemical plume emissions, there is no immediate ground truth or standard to quantify, evaluate and compare the success of different detectors.

2.1. **Objectives.** The ultimate goal of this line of research is to detect and identify chemical plumes with a high level of accuracy. This project investigates many different methods to achieve each of these tasks. Since ground truth about the presence of the chemical plume is not available for this data set, an evaluation of the effectiveness of each method will inherently be somewhat subjective. The emphasis of the discussion of each method will be a qualitative assessment of the results and computational efficiency. This report provides an overview of the problem, a survey of methods for detection and identification, and highlight new avenues for potential research.

2.2. **Three-Layer Model.** The three-layer model is a simple method to describe the different components that comprise the spectral radiance measurement for each pixel in the long wave infrared hyperspectral image. Figure 2 illustrates the different objects, or layers, that contribute to the spectral radiance measurement of the long wave infrared spectrometer.

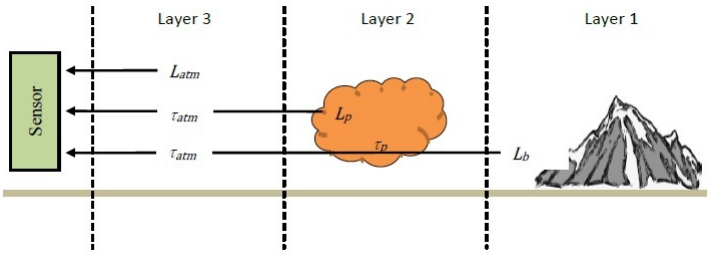


FIGURE 2. Three-layer model depicting the spectral radiance signal received by the long wave infrared sensor.

For the chemical plumes released in this data set, the three layers are the background, the chemical plume and the atmosphere. Each layer has its own radiance $L(\nu)$, and transmittance $\tau(\nu)$. The transmittance is the ratio of light leaving a surface relative to the amount of light entering the medium. Both the background and plume spectral radiances must pass through other mediums before reaching the long wave infrared spectrometer. Therefore, the spectral radiance measurement of the sensor can be represented as

$$L(\nu) = \tau_{atm}(\nu)L_p(\nu) + \tau_p(\nu)\tau_{atm}(\nu)L_b(\nu) + L_{atm}(\nu) \tag{2.1}$$

Before analyzing the hyperspectral data, this model can be further simplified for the given data set. Since the ground-based long wave infrared spectrometers are placed within two kilometers of the chemical plume release site, the spectral radiance of the atmosphere is very small in comparison to the other spectral radiance terms, and therefore can be dropped from this equation. In addition, it is assumed that the atmospheric transmittance does not significantly affect the spectral radiance because of the short path length, allowing most of the signal to pass through. These assumptions reduce the equation into the two layer model equation

$$L(\nu) = \tau_p(\nu)L_b(\nu) + L_p(\nu) \tag{2.2}$$

Thus, according to this model the spectral radiance of the scene measured by the LWIR sensors is a sum of the light emitted by the chemical plume and background mediums at 129 wavelengths in the electromagnetic spectrum.

2.3. **Data Conversion.** The first step in data processing is to convert the hyperspectral data from spectral radiance to emissivity. The emissivity of a surface is a measurement of the light leaving a surface relative to that of a perfect blackbody, ranging between 0 and 1. As described in [3] provided by the Johns Hopkins University Applied Physics Laboratory, Planck's blackbody equation directly relates the spectral response of a surface to emissivity and temperature,

$$B(\nu, T) = \frac{2hc^2\nu^3}{exp(\frac{hc\nu}{kT}) - 1} \tag{2.3}$$

where $B(\nu, T)$ is the spectral exitance at a given wavenumber ν and temperature T, h is Planck's constant, c is the speed of light, and k is Boltzmann's constant. The algorithm for this conversion was provided by the Johns Hopkins Applied Physics Laboratory, and the calculation makes assumptions about atmospheric conditions to convert the data. Specifically, the assumption that the temperature of the entire scene is the same is an inaccurate approximation. The temperatures of the distant mountains in the background, the foreground of the desert into which the plume was released and the emitted chemical plume itself are different temperatures. Due to this approximation, there are resultant outliers in the emissivity data, outside of the expected 0 to 1 emissivity range. To clean the data set, a median filter was implemented on the outlier pixels prior to running the plume detection and segmentation algorithms.

The spectral signature of a given pixel at a specific wavenumber is a mixture of multiple materials at different temperatures in the hyperspectral image. The resultant pixel emissivity can be modeled by the expression

$$\epsilon(\nu) = \sum_{i=1}^{N} \beta_i \epsilon_i(\nu) \tag{2.4}$$

where β_i are the percent of each material present in the pixel, ϵ_i is the emissivity signature for the ith material, ν is the wavenumber and N is the number of material components for a given pixel. The emissivity signature of each material varies with respect to temperature, so that a library of spectral signatures for a particular material varies based on testing conditions including temperature and wavenumber.

3. Filtering/Preprocessing

3.1. **Background subtraction.** Due to the high dimensionality and type of data, the chemical plume emissions are not visible in the raw data of spectral radiance nor in the converted emissivity data. A useful method for preprocessing is to perform background subtraction on the data. For the video sequences of chemical plume emissions, this method is effective because the gaseous plume is the only moving object in the image. Background subtraction calculates the difference between the previous frame of data and the current frame so that the resulting difference creates a disturbance map that identifies motion in the image. This simplistic approach may yield a noisy disturbance map when there is a high degree of variability between individual frames.

As a result, a modified real-time algorithm for background subtraction has been implemented for the Johns Hopkins Applied Physics Laboratory data set. This computation of the disturbance field creates a temporal average image that is a running weighted average of previous frames.

$$\begin{aligned} A_t &= (1-w)I_t + wA_{t-1} \\ \delta_t &= I_t - A_{t-1} \end{aligned} \tag{3.1}$$

The approach gives greater weight to more recent frames and decreases the noise that arises from background subtraction. In the equation shown above, A_t denotes the temporal average at time t, I_t is the current frame of data, δ_t is the disturbance field and w represents the history weight factor, ranging from 0 to 1. After summing along the spectral dimension, a two dimensional disturbance map results in this background subtraction algorithm as can be seen in Figure 3.

The advantages of background subtraction are that this computation is relatively quick and is viable for implementation in real time, making it highly applicable to airborne toxin detection in real world applications. Compared to other segmentation techniques, the calculated disturbance map yields useful results, especially in data sets in which the mountainous background interferes with other segmentation algorithms. However, this method

since it is a motion detector, it is not a robust standalone plume detection technique, especially if there are other types of movement in a given image.

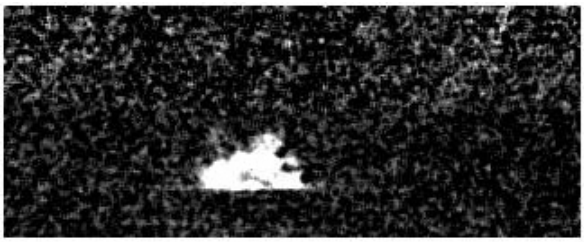


FIGURE 3. Grayscale disturbance map of aa12 Victory chemical plume release.

3.2. **False color representation.** In order to obtain a better visualization of the dataset, principle component analysis (PCA) is used to reduce the dimension of the data from 129 to 3. This means that for each pixel, there are now 3 channels as opposed to the original 129 wavelengths in the spectral dimension. This allows for a false color representation of the data, where the first principal component is labeled to be red, the second component is blue, and third component green. This allows for a color visualization of the data in which the foreground, background and chemical plume.

While this visualization is incredibly clear, there is flickering inconsistency in the RGB display. From frame to frame, the values for each color components shifts due to noise, variance in the data from frame to frame and possible artifacts from the LWIR sensor, causing the color to shift constantly between frames. To remedy this situation, the Midway Equalization method[4] was utilized, a modified version that takes advantage of how the video is taken in discrete time, as opposed to a continuous one. The Midway Equalization method is used to equalize the histogram representation of data.

Algorithm for Midway Equalization[4]

Input: RGB Image

- Load all of the video frames into storage, $F_1, F_2, .., F_N$
- Separate each of the frames into its color components, $F_1 \Rightarrow \{F_{11}, F_{12}, F_{13}\}$
- Sort each colored component frame, $\overline{F_{11}}, \overline{F_{12}}, \overline{F_{13}}$
- Obtain the average of each colored component

$$\begin{array}{rccc} & \overline{F_{11}} & \overline{F_{12}} & \overline{F_{13}} \\ + & \overline{F_{21}} & \overline{F_{22}} & \overline{F_{23}} \\ + & ... & ... & ... \\ + & \overline{F_{N1}} & \overline{F_{N2}} & \overline{F_{N3}} \\ \hline & \tilde{F_{N1}} & \tilde{F_{N2}} & \tilde{F_{N3}} \\ \hline \Rightarrow & \tilde{F_{N1}}/N & \tilde{F_{N2}}/N & \tilde{F_{N3}}/N \end{array}$$

- Replace all of the values in F_{11} with the new values in $\tilde{F_{N1}}/N$, perform this for all the other frames and colors

Output: Equalized Image

Figure 4 shows the resulting application of Midway Equalization. The original video flickers between frames, as shown, each of the 3 original frames have a different coloring, the first is predominantly green, second blue, and third red. After the equalization, each of the 3 images are similar shades of purple and pink. This provides continuity for the video sequence and displays movement of the plume more clearly as the scene changes from frame to frame.

4. Detection

The development of remote sensing technology has allowed scientists to measure the light that is reflected and absorbed in a scene at various wavelengths in the electromagnetic spectrum and determine the materials that are present. Various algorithms have been developed for identifying materials and extracting spectral signatures within an image. The extension of still hyperspectral images to video sequences for this data set increases the amount of computation needed to generate results on a frame by frame basis. For example, extraction of all material signatures present in a scene can be computationally expensive to perform, especially per frame.

In the application of chemical plume detection in video sequences, it is not necessary to perform complete background estimation and unmixing techniques. In our case, we are given a library of the emissivity signatures of the chemical gases collected in a laboratory and know which chemical is present in each video sequence. In a sense, this is a type of semi-supervised unmixing. The task is to identify pixels in each frame that contain traces of the chemical plume, using the provided characteric emissivity signatures.

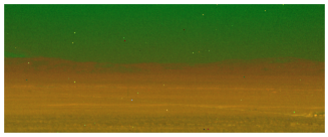
Original Image	Midway Equalized Image
	
	
	

FIGURE 4. Results of the Midway Equalization

4.1. **Estimation of background signatures.**
The estimation of emissivity signatures that are to be classified as background is a process referred to in literature as *background endmember extraction*. For the case of LWIR hyperspectral video, background signatures must be updated at least every few frames. This is a byproduct of the underlying physics and the inherent nature of emissivity signatures. Since the spectral signature of the gas is known, we need to estimate other signatures present in the scene that are different from that of the plume. For each method of endmember extraction, background subtraction was used to determine which pixels to consider. Any nonzero pixels in the disturbance map indicated movement and possible presence of the gas plume, and was therefore excluded from background endmember extraction.

4.1.1. *Automatic Target Generation Process (ATGP).*

Finding pixels that are very distinct from one another will provide a good representation of signatures that are present in an image without any prior knowledge of materials in a scene. This can be done through an iterative process of orthogonal projections. The algorithm is as follows

- Initialize with a random pixel, ϵ_0
- Find pixels ϵ_i, such that
  $\epsilon_i = \arg\max_x \|(I - U_{i-1}(U_{i-1}^T U_{i-1})^{-1} U_{i-1}^T))x\|_2$
  where $U_{i-1} = [\,\epsilon_1\, \epsilon_2\, \ldots \epsilon_{i-1}\,]$ is the subspace spanned by the previously found distinct pixels, and x is a pixel in the image.

Note that the initial random pixel, ϵ_0, is not retained in this process. The purpose of the orthogonal projections is to identify pixels that are not well represented in the $span\{U_{i-1}\}$. These pixels are considered distinct, and stored as another column in U_i. This procedure will show preference to pixels that have very large magnitude, so normalization is required before forming these orthogonal projections.

4.1.2. *Principle Components Analysis (PCA).*
Treating the image A as a data matrix where columns correspond to pixels and rows correspond to spectral bands, the principle components transform may be found by creating the covariance matrix AA^T and computing eigenvalues and eigenvectors. The resulting eigenvectors correspond to directions in which the data set A has the greatest variance. That is, the first principle component (the eigenvector corresponding to the largest eigenvalue) captures the direction that captures the most variance within the data. In the case of a hyperspectral image these directions are the dominant spectral signatures.

4.2. **Adaptive Matched Subspace Detector (AMSD).**
The Adaptive Matched Subspace Detector is a probabilistic detection scheme that uses a generalized likelihood ratio test to choose between the hypotheses

$H_0 : x = S_b u_b + n$ (Target absent)
$H_1 : x = S_t u_t + S_b u_b + n = Sx + n$ (Target present)

where $n \sim \mathcal{N}(0, \sigma_w^2 I)$, x is a pixel in the image, S_t is a matrix of target signatures, S_b is a matrix of background signatures, and u is a vector of abundances of the spectral signatures in S. The first hypothesis, H_0, corresponds to the situation when the pixel x may be represented by only background pixels plus noise. The hypothesis H_1 says that the target signature is needed in addition to the background to fully represent the pixel. A

hypothesis test is constructed using a generalized likelihood ratio, given by

$$\mathscr{L}(x) = \left(\frac{x^T P_b^{\perp} x}{x^T P_S^{\perp} x}\right)^{L/2} \tag{4.1}$$

where, the matrix P_S is the projection onto the subspace spanned by S. To make compare $\mathscr{L}(x)$ to a given threshold ℓ_0, and pick H_0 if $\mathscr{L}(x) < \ell_0$ and H_0 otherwise. In order to ensure linear independence (that is uncorrelated) of the numerator and denominator the ratio

$$T_{AMSD}(x) = \frac{x^T(P_b^{\perp} - P_S^{\perp})x}{x^T P_S^{\perp} x} = \mathscr{L}(x)^{L/2} - 1 \tag{4.2}$$

is used. Further information on the probability distribution associated with equation (4.2) may be found in [6].

It is noted in the literature that these types of detectors have a high rate of false alarms. Also, this implementation used PCA as a means of background endmember extraction. As mentioned previously, PCA occasionally returns signatures close to the signature of the plume. Both of these issues contribute to noise in the final result.

4.3. L_1 Unmixing.

The purpose of unmixing is to calculate the abundance of particular emissivity signatures that each pixel is made up of. So for a given $m \times n$ matrix A where columns are spectral signatures, the solution of

$$\begin{gathered} \min_u \|Au - f\|_2^2 + \eta|u|_1 \\ \text{s.t. } u \geq 0 \end{gathered} \tag{4.3}$$

where f is a pixel in the image, will result in u giving the abundance of each signature in A. The parameter η controls how sparse the solution is. Note that if sparsity is not desired in the solution (ie $\eta = 0$), this becomes a non-negative least squares problem. The minimization problem (4.3) is solved numerically using a particular type of Split Bregman iteration detailed in [5]. It is optimized for solving large numbers of small to medium overdetermined problems (when $m > n$) with the same A, which results in faster performance than MATLAB's non-negative least squares solver.

4.4. Results.

In order to provide an idea of how well these algorithms perform under different circumstances, the results of each will be presented for two different stages of plume diffusion.

The AMSD detector had a very strong response to the plume when the cloud is highly concentrated. This can be seen in the top image of figure 4.4 by the presence of red pixels in the area of the chemical release. Over the course of the next several frames the plume diffuses quite rapidly. The

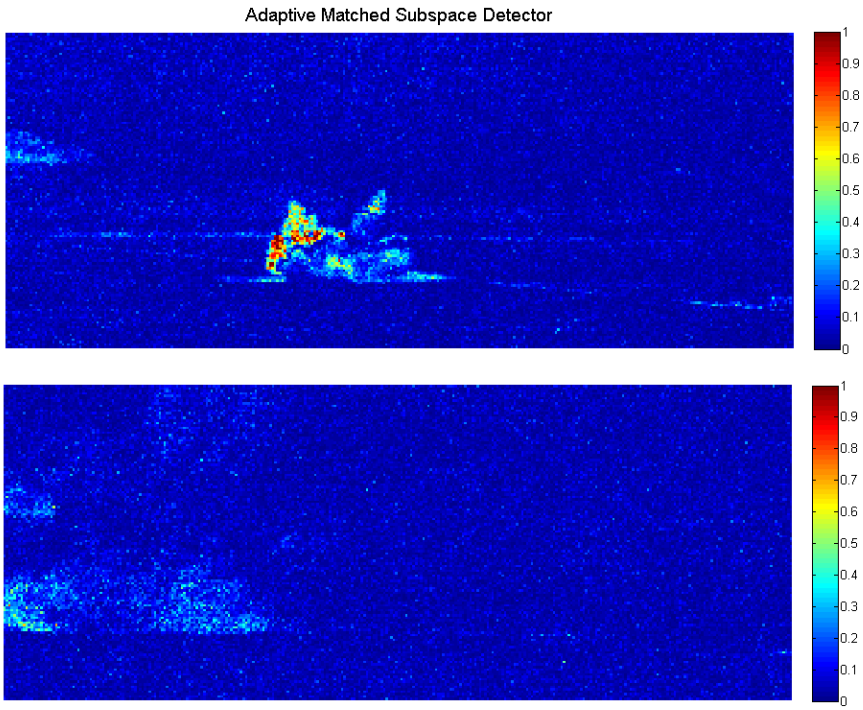


FIGURE 5. Results of the AMSD algorithm. The top image is an early frame in the sequence, before the plume had time to diffuse. The bottom image is later in the video sequence, after the plume has become thin.

AMSD detector results begins to detect quite a bit less plume over these frames, as can be seen in the bottom image of 4.4.

It is interesting to note that the background signatures selected by both the ATGP and PCA process are very similar. The results of unmixing before the plume diffuses significantly may be seen in figures 4.4 and 4.4. When the plume becomes more diffuse the L_1 unmixing gets a stronger response for the desired target signature when the background endmembers are selected by the ATGP process. This can be seen by comparing 4.4 to 4.4 and noting the intensity of the response in the top left image. Also notice that both ATGP and PCA selected an endmember signature that was close to the signature of the desired target.

4.5. **Remarks.**

When a chemical plume diffuses into the surrounding environment, the estimation of background signatures without interference of the plume signature is a challenge. Incorporating temporal information can improve consistency

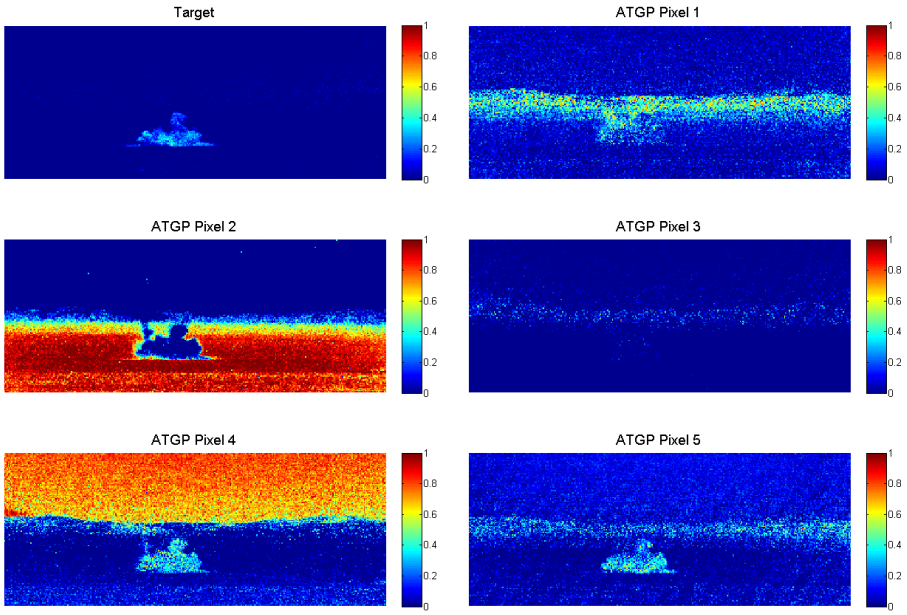


FIGURE 6. Results of L_1 unmixing with selecting background endmembers by ATGP. Notice the clear detection of the plume in the upper left image.

considerably. For example, taking into account 5-10 previous disturbance maps and ignoring all pixels that changed over these frames accounts for the dynamic motion of the plume, particularly since the frame rate is 0.2 Hz. However, when the plume becomes optically thin it becomes harder to identify and ignore pixels that may be contaminated by its signature. In this case it is much more difficult to guarantee that the estimated background signatures are independent of the signature of the plume. Both PCA and ATGP produced background signature estimates that were considered plume by L_1 unmixing. The principle components change quite drastically between frames in LWIR hyperspectral video sequences. It is also worth noting that PCA was particularly sensitive to producing estimates close to the the spectral signature of the plume if care is not taken to ensure the remaining pixels have not been contaminated. Again, temporal information may by used to improve these estimates. This also had the advantage of improving consistency of the principle components of consecutive frames.

5. SEGMENTATION

5.1. **Introduction.**

For our program, we utilized segmentation algorithms in order to try to isolate the gas plume. Segmentation is the partitioning of data into clusters, where each cluster represents a different element of the data. This process has been utilized in image processing, where each cluster represents a subject

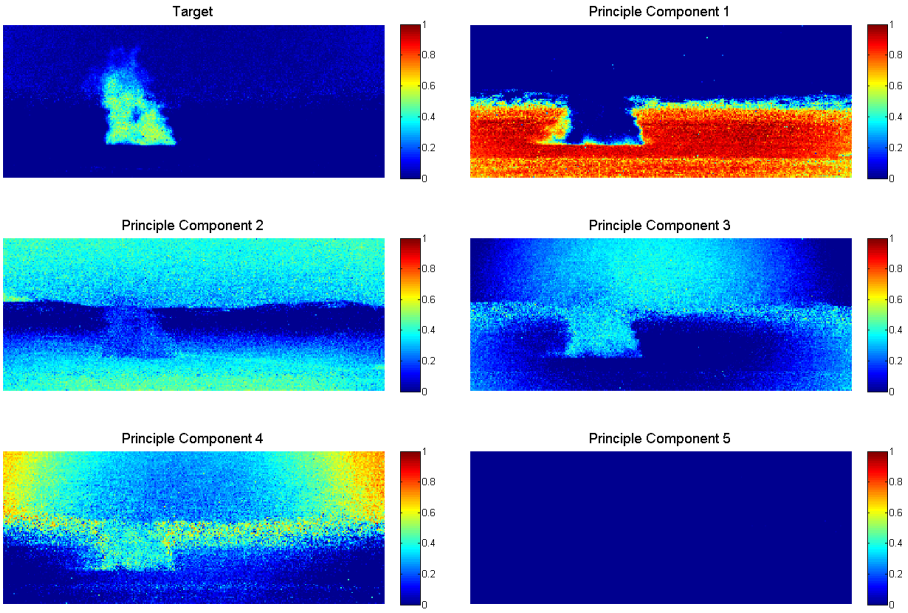


FIGURE 7. Results of L_1 unmixing with selecting background endmembers by PCA. Notice the clear detection of the plume in the upper left image.

in the image. We chose to perform image segmentation on our data because we were looking to separate all of the distinct components. The advantage of segmentation algorithms is that they are able to work

Like other AI algorithms, there are supervised, semi-supervised, and unsupervised methods of segmentation. A supervised method is one where the user would do all the work, deciding if a pixel in an image belongs to a person or if its a piece corn or etc. A semi-supervised method requires user input for a few points, after which the algorithm would make decisions on the remainder of the points. An unsupervised method requires no user input and makes the determination of each cluster based solely on the data. The ideal situation would an unsupervised method that is able to isolate the gas plume, because we want the process to be automated.

Overview of segmentation methods		
Method	Supervision	Average Run-time
K-Means	Unsupervised	<1 min
Spectral Clustering	Semi-supervised or Unsupervised	5 min
Nyström	Unsupervised	<1 min
Ginzberg-Landau minimization	Semi-supervised	1<2 min

A distance metric is required when utilizing these segmentation algorithms. We decided to use the the Euclidean and Cosine distance.

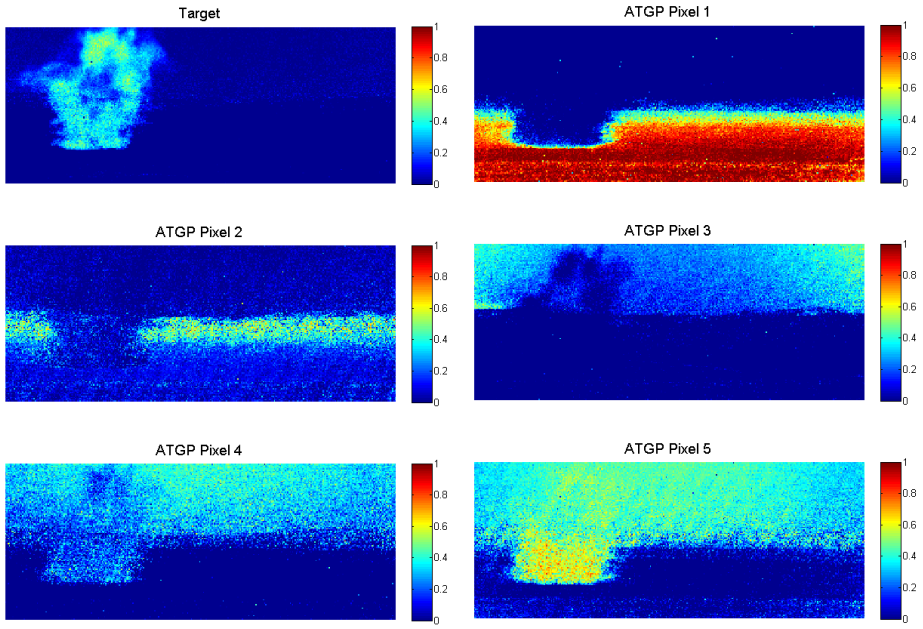


FIGURE 8. Results of L_1 unmixing with selecting background endmembers by ATGP. .

Overview of the different distance metrics $x = (x_1, x_2, ..., x_n)$, $y = (y_1, y_2, ..., y_n)$	
Euclidean distance	$d(x,y) = \| x - y \| = \sqrt{\sum_{k=1}^{n}(x_i - y_i)^2}$
Cosine distance	$d(x,y) = \| x - y \| = 1 - \frac{\langle x,y \rangle}{\|x\|\|y\|}$

5.2. K-Means.

K-means is considered a basic method of segmentation and is usually one of the first methods attempted on a new data set. It is an iterative algorithm that attempts to segment the data into K clusters.

Algorithm for K-means

Input: Datacube

(1) Initialize k centers, labeled $C_1, C_2, ..., C_k$
(2) Calculate the **distance** between each point and each center
(3) Classify each point with a label, associated with the closest center
(4) Calculate the mean of the points in each cluster, and set that as the new center
(5) Go to step (2) and repeat this process until the points do not change clusters

Output: Classification for each data point

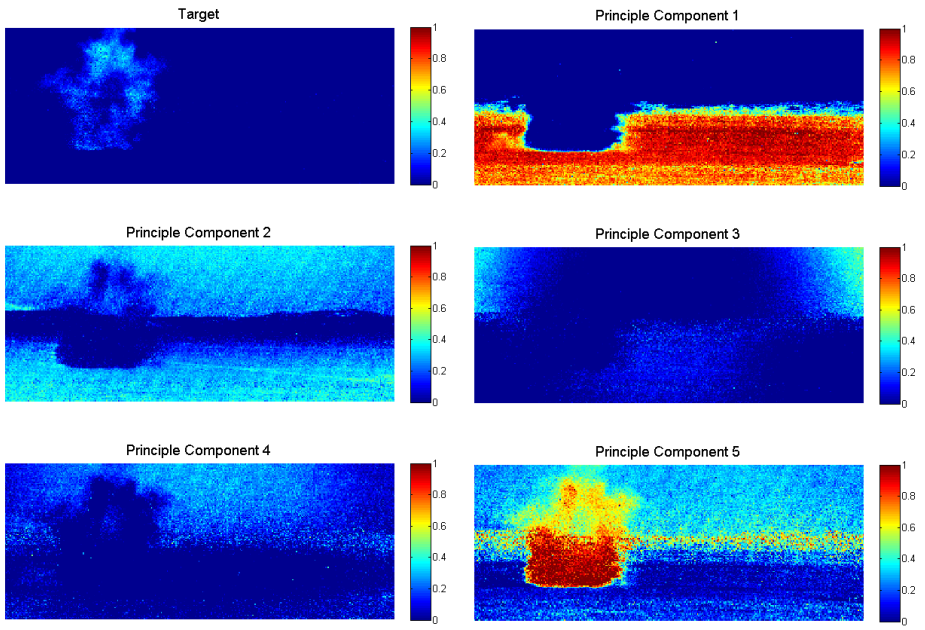


FIGURE 9. Results of L_1 unmixing with selecting background endmembers by PCA.

The algorithm has three areas where adjustments can be made, the number of centers, initialization of centers, and the chosen distance metric. The number of centers determines how many clusters we formulate. Choosing k too large will result in trivial clusters that contain only one point and could potentially break up clusters. Choosing k too small will result in a clusters that are larger than they should be. Ideally, k is chosen such that it is the same as the number of elements in an image. An example is a picture with three distinct objects. Figure 10 shows the different effects of applying k-means with different cluster sizes. It illustrates the biggest problem that arises from applying k-means. If the correct number of clusters is not chosen, the groupings become erratic and gives incorrect clusterings.

This shows the problem that comes when utilizing k-means. However it is a very easy useful algorithm that will give information about the working of a data set. Figure 11 shows some resulting k-means clustering utilizing different distance metrics and k but the same starting centers. In the first frame, there are 4 clear clusters, each different color is part of a different cluster. The upper atmosphere (blue), lower atmosphere (brown), mountain and plume (orange), and foreground (green). There are singular pixels that are dark blue, resulting from outliers. Though none of the resulting clusters shown in Figure 11 are able to capture the gas plume, a few very crucial pieces of information is obtained. The left plots utilize the cosine distance,

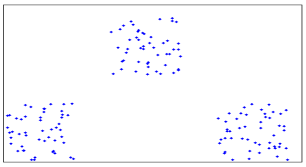
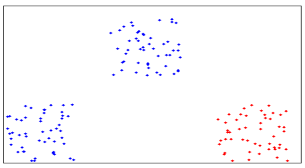
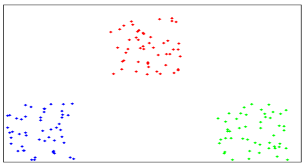
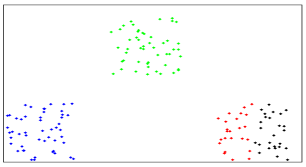
k clusters	Graphics	Comment
$k = 1$		Initialization of the data, pre-clustering. Note the 3 distinct clusters.
$k = 2$		Performing k-means with 2 clusters. The blue cluster has 2 of the groups in one cluster.
$k = 3$		Performing k-means with 3 clusters on 3 clusters. This is correct segmentation.
$k = 4$		Performing k-means with 4 clusters. The bottom left cluster gets split in two.

FIGURE 10. Sample of k-means

while the right plots utilize the euclidean distance. The plots on the left are much more clear than the plots on the right. This tells us that the cosine distance is much more suitable for our data than the euclidean. Another key piece of information that is gleamed is that the emissivity signature of the gas plume are very similar to those of the lowest level of atmosphere

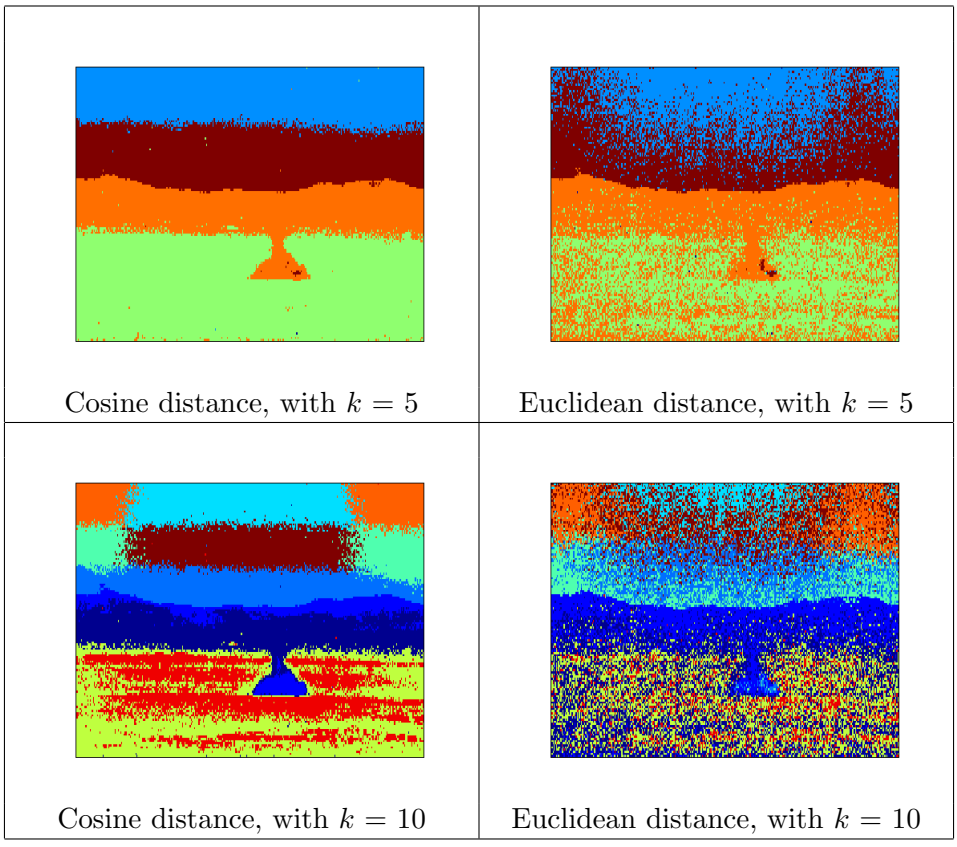


FIGURE 11. K-mean results on Emissivity

and the mountain. And lastly, there are outliers that exist in the data that may play a factor in clustering. Playing around with the different values for k and adjusting the starting centers do not yield any radically differing results.

5.3. **Spectral clusting.**

Spectral clustering is another method of segmentation. The advantage of spectral clustering as opposed to k-means is that it doesn't try to force any number of clusters. While it is very computationally simple, it is much more time consuming then k-means. An overview of the algorithm found in Luxborg[8] is given by:

Algorithm for Spectral Clustering[8]
Input: Datacube • Construct a fully connected similarity matrix • Convert into the Laplacian matrix • Calculate the normalized symmetric Laplacian • Compute the smallest eigenvectors and eigenvalues **Output**: Eigenvectors that represent different clusters and the associated eigenvalues

This algorithm has a few problems that arise due to the nature of the hyperspectral data set. Each image consists of 128*320 frames, so constructing a full similarity matrix is not possible. There is also a problem in computing the smallest eigenvectors and eigenvalues of a matrix. The implemented algorithm in matlab is very time consuming, many times longer than computing the largest eigenvectors and eigenvalues.

Algorithm for Spectral Clustering[7]
Input: Datacube • Construct a similarity matrix of the k nearest neighbors • Compute the normalized similarity matrix • Calculate the largest eigenvectors and eigenvalues of the normalized similarity matrix **Output**: Eigenvectors that represent different clusters and the associated eigenvalues

5.3.1. *Similarity Matrix*[8].
The similarity value is a number representing how closely related two points in $\mathbb{R}^n$ are to each other, 0 means there is no relation and 1 means they are identical. An example of the similarity function is the Gaussian similarity function (5.1). This function utilizes the distance metrics previously discussed, euclidean or cosine. There is also a σ constant, that is used to determine the Gaussian neighborhood. The larger the value of σ the more connect the graph is, culminating in a $\sigma = \infty$ where the similarity function is always 1. Vice versa, the smaller the value of σ the more disconnected the graph is, peaking when σ is 0, resulting in a graph that has completely disconnected from all other points.

$$S(x_i, y_j) = e^{\frac{-d(x_i,y_j)^2}{2\sigma^2}} \tag{5.1}$$

The evaluation of the equation gets placed in (i, j) for the similarity matrix.

The alternative to evaluating all of the similarities and constructing a fully connected similarity matrix is $k$nearest neighbors or ϵ neighborhood. k nearest neighbors is only using the k largest similarities of each row, as

opposed to finding all N similarities. The ϵ neighborhood is using all points that are larger than a set similarity. The results of spectral clustering with these 2 alternative similarity matrices may not be as good, however certain advantages are obtained. Using either of the two will result in a sparse matrix, something that is required for this data set. The k nearest neighbors similarity matrix allows for a customization on sparsity, whereas ϵ neighborhood does not allow for strict controls on sparsity. Due to size constraints, the k nearest neighbor was selected.

Aside form the standard Gaussian similarity function, there is a Self Tuning Similarity function(5.2) that tries to maintain local scaling. The σ_i is the k^{th} nearest neighbor to x_i and σ_j is the k^{th} nearest neighbor to x_j.

$$S(x_i, y_j) = e^{\frac{-d(x_i,y_j)^2}{\sigma_i \sigma_j}} \tag{5.2}$$

Experiments were done to find the best combinations for σ, k, and similarity function.

5.3.2. *Results.*

So now onto results. Figure 12 shows the resulting eigenvalues that come from spectral clustering. The resulting eigenvalues give a lot of information about the behavior of the clusters that are present in the eigenvectors. The values plotted are the result of an eigenvalue/eigenvector solver that arrange the eigenvalues in non-decreasing order. The fact that the eigenvalues present are not in non-decreasing order tells that normalized similarity matrix might have some peculiarities. Looking at the self tuning cosine plot, it appears to not converge to a solution, showing a possibly very ill conditioned matrix. The rate of descent also gives information about the number of trivial clusters that arise. In every image, there is a user perceived number of clusters that are present, these represent non-trivial clusters. In the ideal situation, the eigenvalues should be 1 for all the important clusters, shooting downward to 0 for all the trivial clusters. An example of a trivial cluster would be a singleton point being clustered and the rest of the image in another cluster, seen in the bottom right in figure 13. Because the self tuning cosine has so many clusters at a 1, it shows that there are a number of trivial clusters. Unfortunately due to the nature of the erratic eigenvalues, the resulting eigenvectors are not reliable. The other 3 distance metrics, self tuning euclidean, euclidean, and cosine, all turned out nicely, they each have a gradual decreasing of values which translates to more non-trivial clusters than the self tuning cosine.

Figure 13 shows some of the resulting eigenvectors of the self tuning cosine. As mentioned previously, there are a number of trivial clusters, witnessed by the eigenvalues, and the results are not reliable. Figure 14 shows

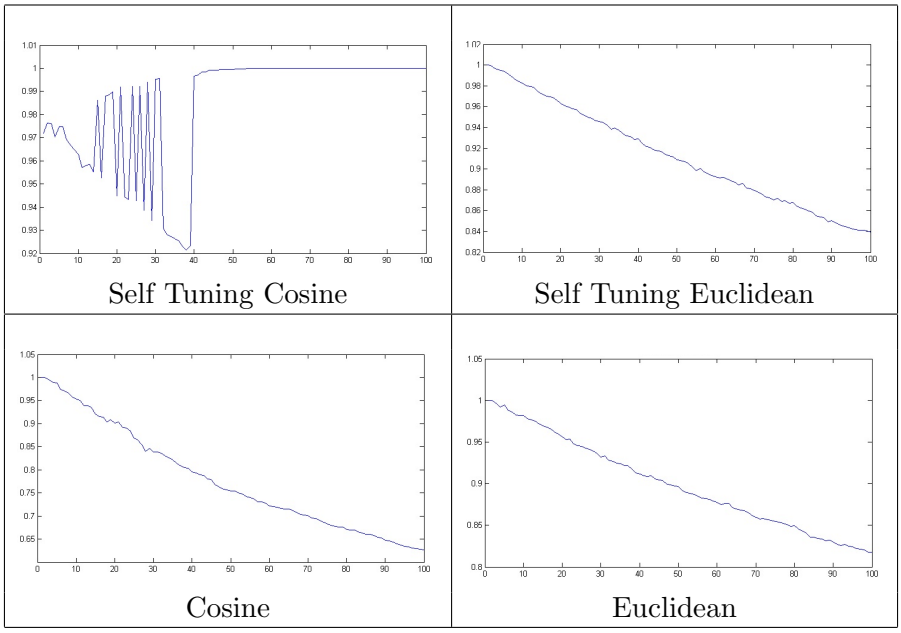


FIGURE 12. Eigenvalues of Spectral Clustering with Different Distances

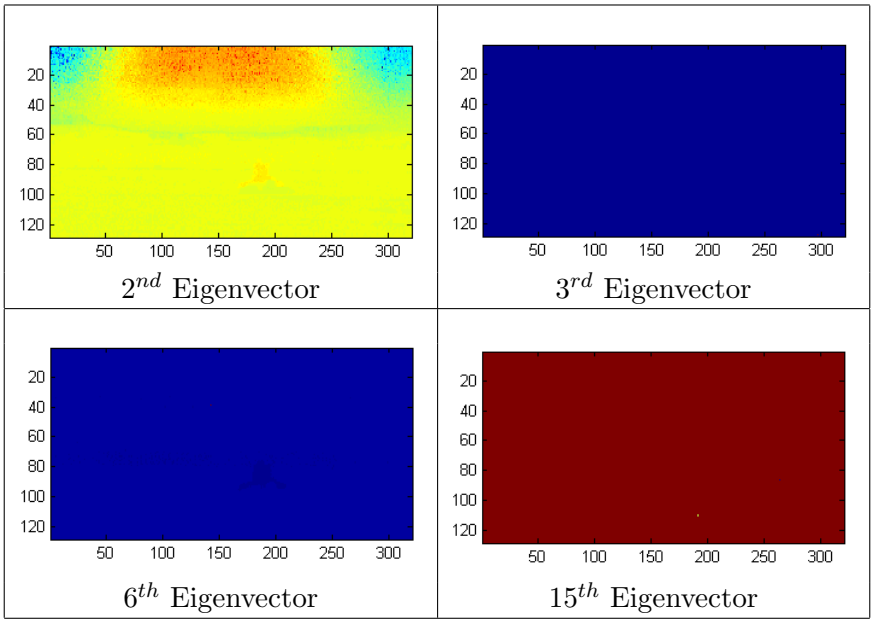


FIGURE 13. Eigenvectors of Spectral Clustering with Self Tuning Cosine

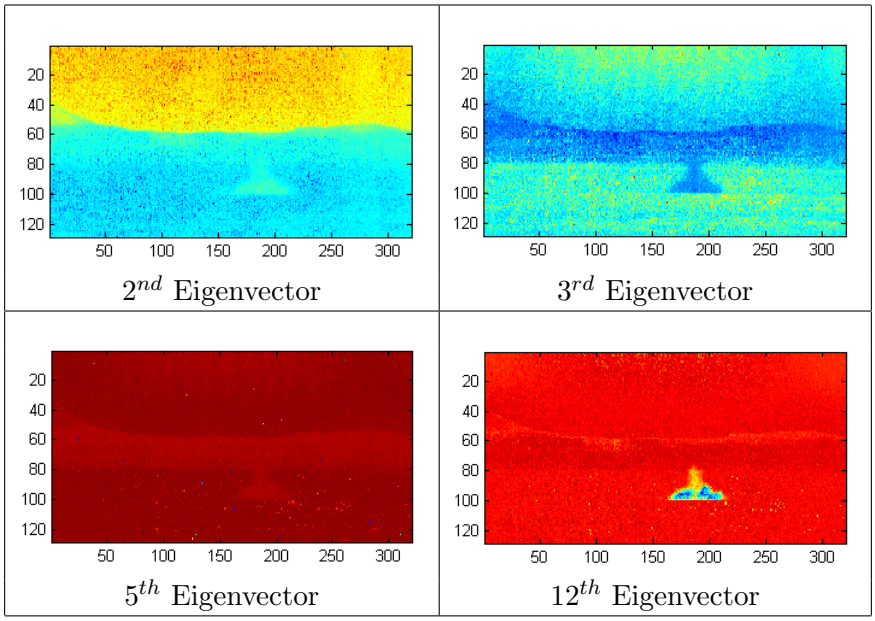


FIGURE 14. Eigenvectors of Spectral Clustering with Self Tuning Euclidean

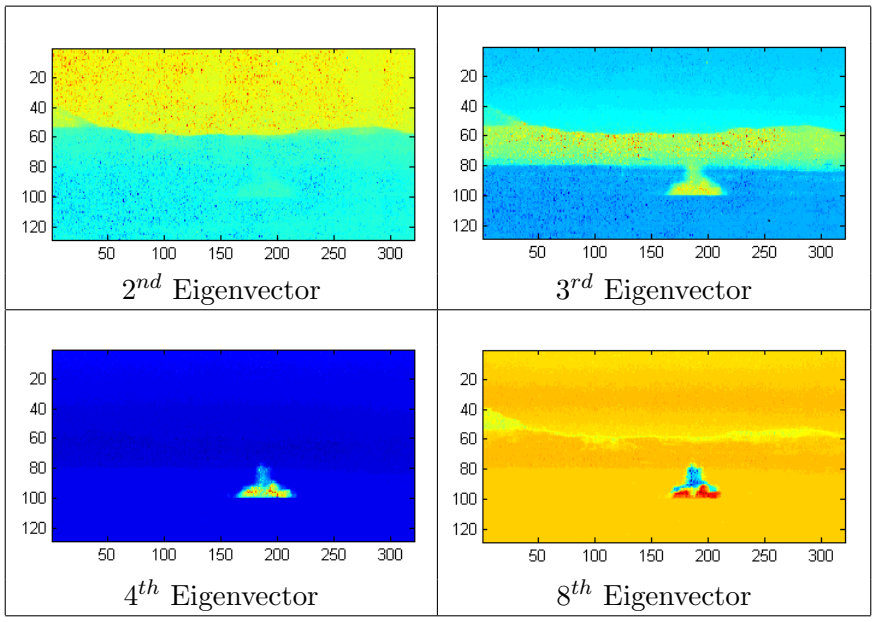


FIGURE 15. Eigenvectors of Spectral Clustering with Cosine

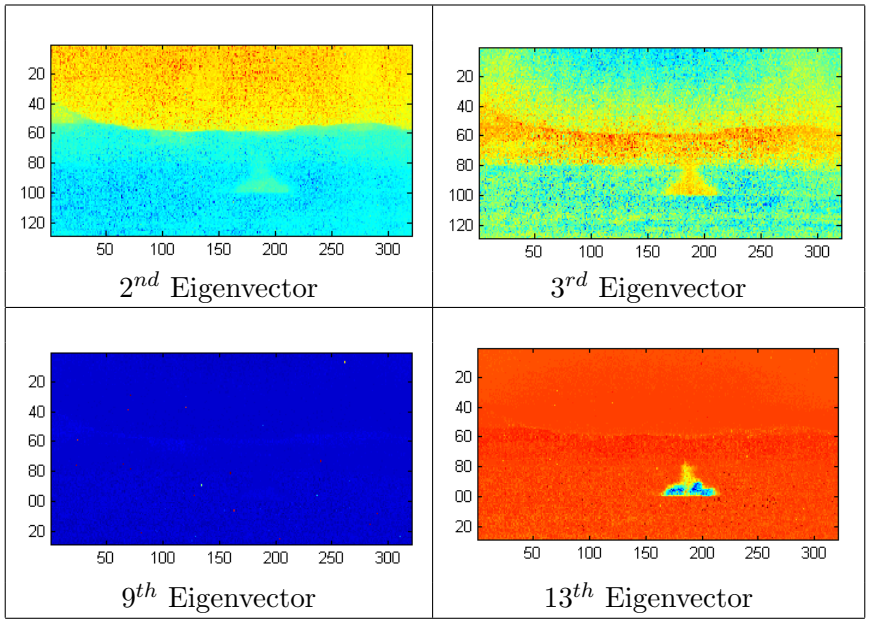


FIGURE 16. Eigenvectors of Spectral Clustering with Euclidean

some of the resulting clusters of the self tuning euclidean. These are much more reliable results because the eigenvalues behaved much more normally. The second and third eigenvectors are shown to demonstrate the predominant clusters that are formed. The second eigenvector separates the sky from the mountain and the third separates the mountain and plume from the other elements in the picture. The fifth eigenvector is an example of how outliers get clustered together. The twelfth eigenvector is the desired plume, plume in one cluster and everything else in the other cluster. One thing to note is how the clustered areas are not clear and distinct, there are many fuzzy areas. These results are quite similar to those shown by k-means. Figure 16 shows the results of the standard euclidean distance. The second and third eigenvector are similar save for slight variations. The ninth eigenvector is another example of outliers being clustered together and the thirteenth eigenvector shows the proper clustering of the plume. Again the fuzzy clusters are seen, similar to those of the k-means and spectral clustering with self tuning euclidean. Lastly is figure 15 with the best results yet. Each of the clusters have a clearly defined border, and the desired gas plume is seen as early as the fourth eigenvector. However, an interesting cluster is seen in the eighth eigenvector. It has a different shape from the fourth eigenvector, however it inhabits the same area, giving rise to the notion that it might be picking up something different from the gas plume, possibly dust that was kicked up from the initial explosion.

The final verdict is that the cosine distance metric is best in the clustering of data in both k-means and spectral clustering.

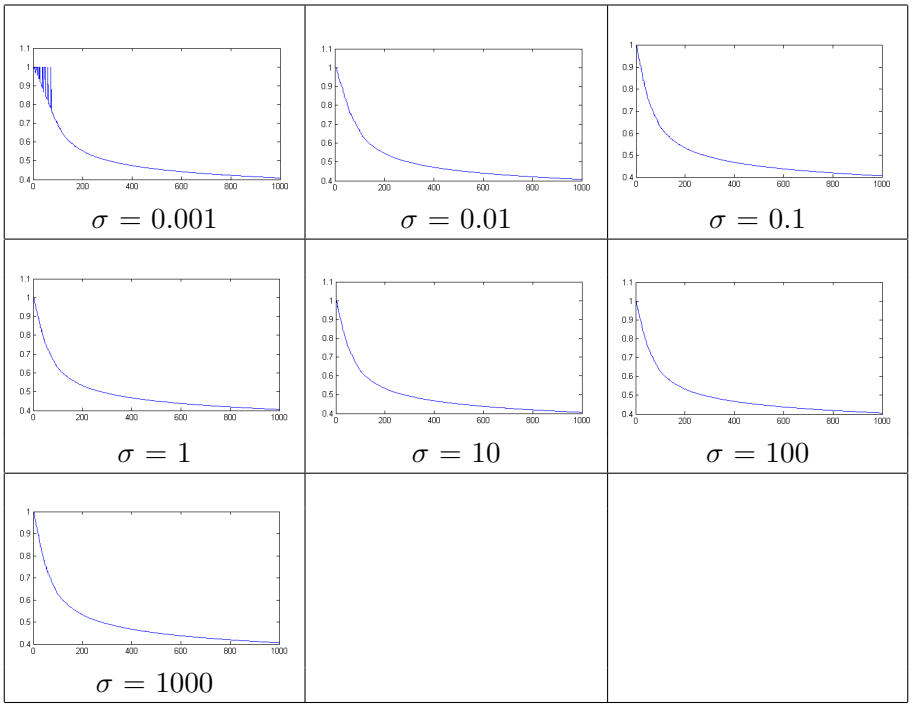


FIGURE 17. Eigenvalues of Spectral Clustering using Cosine and different σ

Next is an examination on the different values of sigma and different sizes of k for the k nearest neighbors in the similarity matrix. Figure 17 shows the resulting eigenvalues for different sigma and figure 18 shows the eigenvalues for different values of k. Due to the wild perturbations in the eigenvalues for $\sigma = 0.001$, this gives rise to the same belief as the self tuning cosine and how it has many trivial clusters. This gives the hypothesis that smaller values of σ gives trivial clusters, which follows the idea that smaller σ values forces the points to become disjoint. Examinations of the eigenvectors show that values of σ aside from 0.001,0.01 seem to give the plume cluster and reduce the trivial clusters. Investigations of the k nearest neighbors gives interesting results. It is believed that the more connections you draw between points the better the clustering, however there are high frequency oscillations in later eigenvalues which means the corresponding eigenvalues are not dependable.

The resulting eigenvectors from spectral clustering with different values for k is shown in figure 19. Only the clusterings with plume are shown, however each of the eigenvectors were within the first 8 eigenvectors. As the number of k increases, the clusterings of the plume start to degrade. This is due to the fact that as the number of connections increase, the more connections that are made. For the cliassification of plume, it seems best to use a k between 10 to 20.

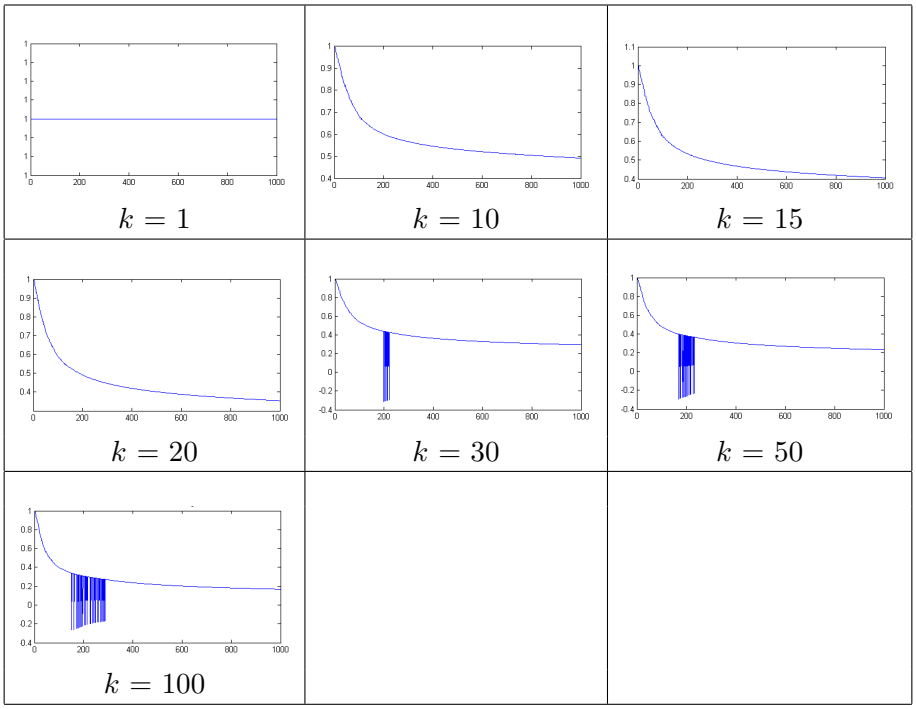


FIGURE 18. Eigenvalues of Spectral Clustering using Cosine and different k

5.4. **Nyström**[2]**.**

Due to the time consuming nature of full spectral clustering, alternatives were looked at. Nyström method for approximating the eigenvalues and eigenvectors of a matrix is a very fast algorithm that utilizes a random sampling in order to perform the computations. Quite reminiscent of k-means, the initialization of the sampling can change the outcome of the computation, however increases the number of samples taken can reduce the variations in results. The algorithm, outline below, is quite similar to that of full spectral clustering, however time is saved because you do not need to compute the k nearest neighbors, which ultimately requires you to find all of the distances between each of the points. Analysis of the eigenvalues and eigenvectors should be exactly the same as in full spectral clustering.

Figure 20 shows the different eigenvalue approximations and figure 21 shows the eigenvector approximations. The eigenvalues are different from the spectral clustering eigenvalues, in that it is non-decreasing whereas spectral clustering results in non-increasing eigenvalues. The analysis is still the same. The incredibly steep increase of the eigenvalues shows that of the approximated eigenvectors, only a few will contain non-trivial clusters. An evaluation of the different sample size yields that there is very little difference between sample sizes of 10, 40, and 100. However, the σ values run contrary

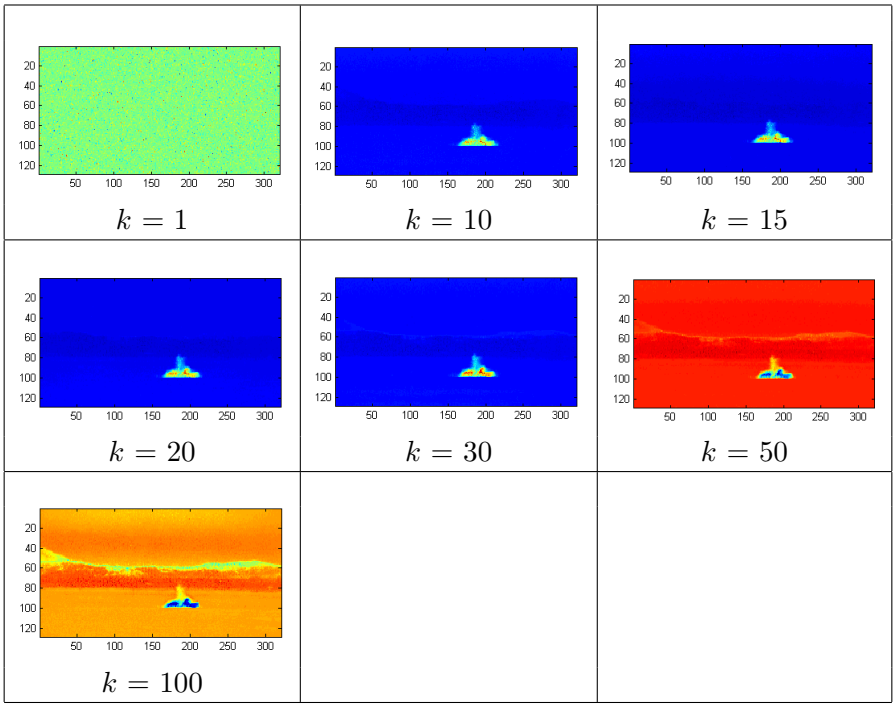


FIGURE 19. Eigenvectors of Spectral Clustering using Cosine and different k, only the eigenvectors with plume

to the results in full spectral clustering, the smaller values of σ result in a more detailed image than the larger values. The results from Nyström are ultimately not useful as a primary means of segmentation. The number of non-trivial eigenvector clusterings is not enough to be able to isolate the plume from the background. But, the results show promise when used as a precursor for other methods, like the Ginzberg-Landau minimization.

Algorithm for the Nyström method[2]
Input: Datacube • Select a random sample of the data points • Construct a similarity matrix only using the random selection • Construct a simliarity matrix between the selected points and the remainining points • Normalize the constructed similarity matrices • Calculate eigenvectors and values **Output**: Eigenvectors that represent different clusters and the associated eigenvalues

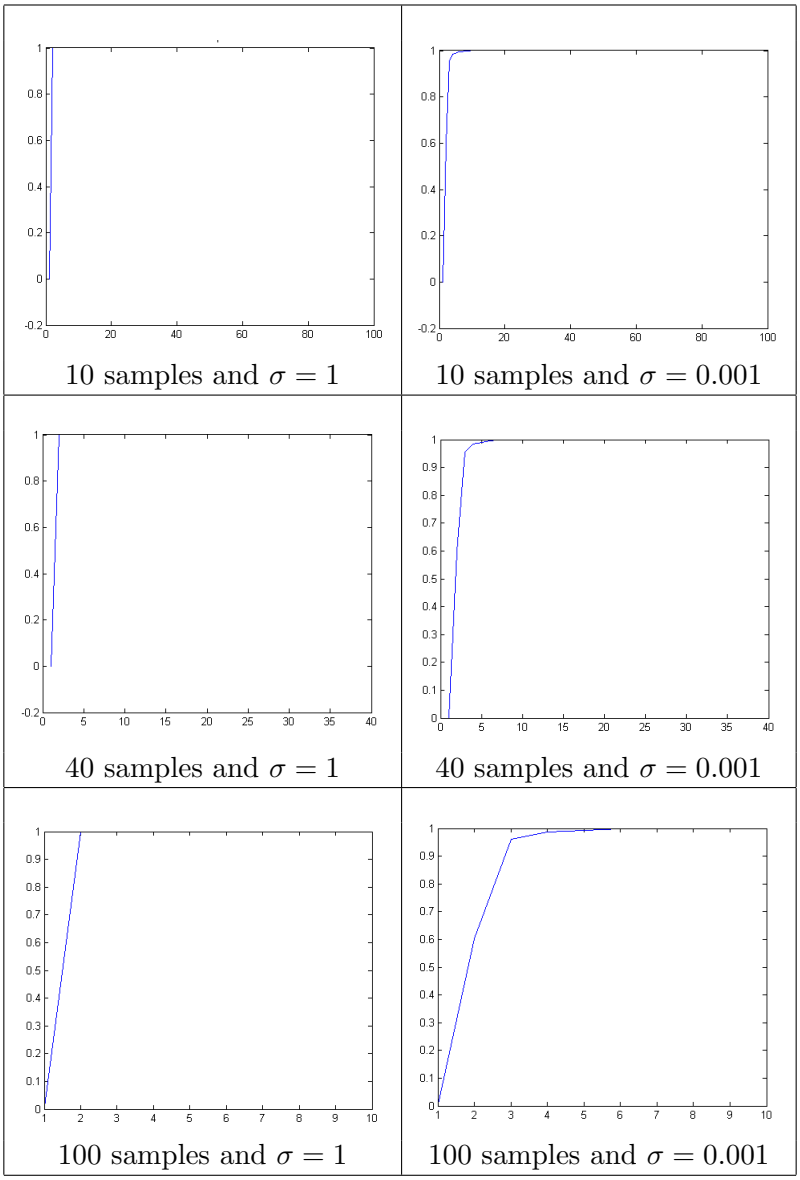


FIGURE 20. Eigenvalues resulting from Nyström method

5.5. **Ginzberg-Landau minimization.** The Ginzberg-Landau functional is

$$GL(u) = \frac{\epsilon}{2}\int |\nabla u|^2 dx + \frac{1}{\epsilon}\int W(u)dx \tag{5.3}$$

where $W(u)$ is a double well potential, such as $W(u) = (u^2 - 1)^2$. When minimizing (5.3) the double well potential term will force solutions to the

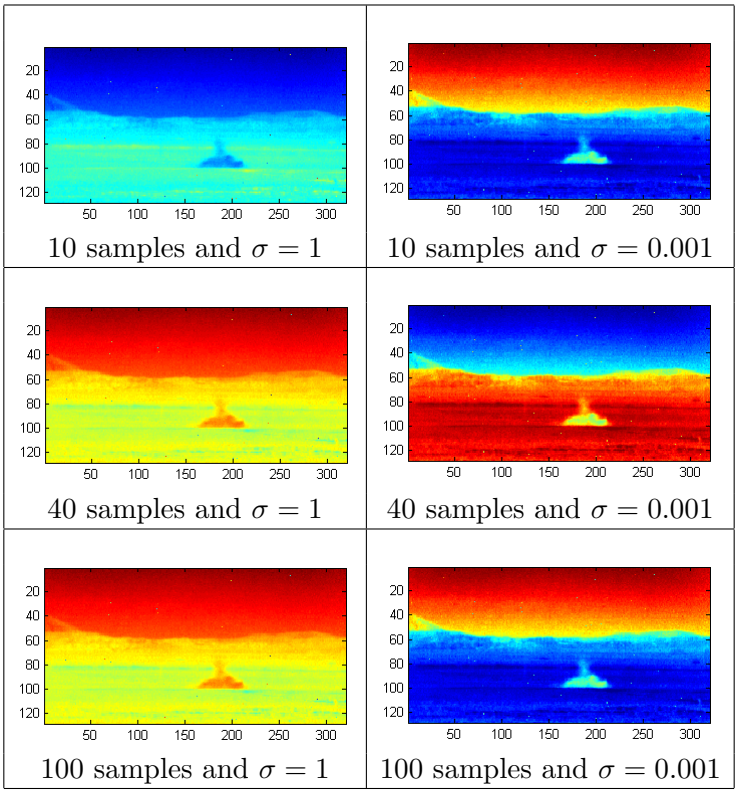


FIGURE 21. Eigenvectors resulting from Nyström method

minimizers of $W(u)$. The first term in equation (5.3), containing the spatial gradient operator ∇, will incorporate smoothness into the solution. Therefore, any sharp transitions between the two minimizers of $W(u)$ will be smoothed out. The solution that minimizes (5.3) will have regions close to one of the minimizers of $W(u)$, as well as an interface between the two regions. Models with this property are referred to as "diffuse interface" models.

In order to use this technique to segment an image, this method is modified to work on a graph. In order to do this the gradient term is replaced with a non-local, graph version of the discretized Laplace operator, $\epsilon u \cdot L_s u$, and a fidelity term is added. The resulting functional is,

$$GL(u) = \epsilon u \cdot L_s u + \frac{1}{\epsilon} \int W(u) dx + \int F(u, u_0) \tag{5.4}$$

where $F(u, u_0)$, is the additional fidelity term. This method is a semi-supervised segmentation algorithm and is initialized with a patch of the region of interest, u_0.

5.6. **Minimization Techniques.**

5.6.1. *Bertozzi-Flenner.*

The functional (5.4) is minimized by applying the method of gradient descent to

$$\frac{\partial u}{\partial t} = -\epsilon L_s u - \frac{1}{\epsilon} W'(u) - C\lambda(x)(u - u_0) \tag{5.5}$$

and evolving (5.5) to steady state by convex splitting methods. Note that the fidelity term was taken to be $\frac{1}{2}C\lambda(x)(u-u_0)^2$. For a more detailed explanation of the method see [2].

When running this method there are five parameters in the numerical scheme:

(1) C: parameter in the convex splitting
(2) ϵ: interface width parameter in the G energy
(3) c_1: parameter in front of the fidelity term
(4) Δt: time step
(5) N: number of iterations in energy minimization

5.6.2. *Modified MBO Scheme.*

The paper by Merkurjev et al. proposes a modification of the MBO (Merrimen, Bence, and Osher) scheme in order to minimize the Ginzberg-Landau functional [REF]. The method is a two step process. The first step is solving a heat equation of a graph, and the second step is thresholding. More specifically,

(1) $y(x) = S(\delta t)u_n(x)$, where $S(\delta t)$ is the evolution operator of the equation

$$\frac{\partial z}{\partial t} = -L_s z - C_1\lambda(x)(z - z_0)$$

(2) Threshold,

$$u_{n+1}(x) = \begin{cases} 1 & \text{if } y(x) \geq 0 \\ -1 & \text{if } y(x) < 0 \end{cases}$$

This is repeated for a prescribed number of iterations, or until $\frac{\|u_{new}-u_{old}\|_2^2}{\|u_{new}\|_2^2} < 10^{-6}$ is satisfied. A full discussion of this method may be found in [1].
This method requires fewer parameters than the Bertozzi-Flenner method. The convex splitting term, C, disappears entirely, and ϵ get absorbed into Δt.

5.6.3. *Results.*

The Bertozzi-Flenner method took between one and two minutes per frame and typically required approximately 400 iterations for the pixels in the image to converge to 1 or -1. Using the smoothed disturbance map found

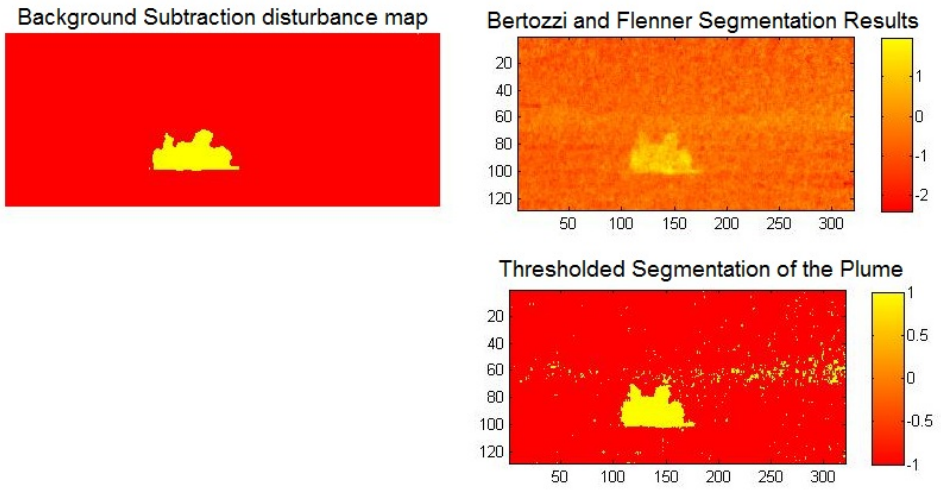


FIGURE 22. (Top left) Disturbance map resulting from background subtraction. (Top right) Disturbance map after median filtering to remove noise. This was used to initialize the Ginzberg-Landau segmentation (Bottom left) Results of the Bertozzi-Flenner method

from background subtraction, this method effectively segments the plume and detects the wisps at the edges of the diffusing gas. The algorithm is highly dependent on the parameter values and the best results were obtained with c1=100, c=5, ϵ=1 and dt=0.001.

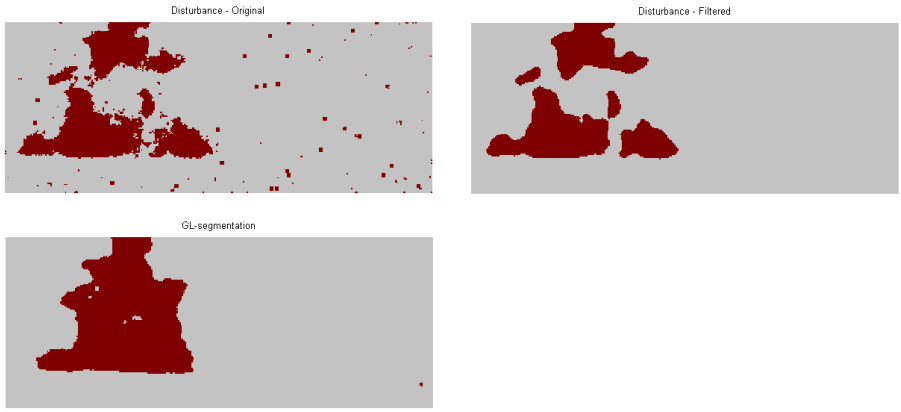


FIGURE 23. (Top left) Disturbance map resulting from background subtraction. (Top right) Disturbance map after median filtering to remove noise. This was used to initialize the Ginzberg-Landau segmentation (Bottom left) Results of the Merkurjev et al. MBO scheme

The Merkurjev et al. MBO scheme on average took between 5 and 10 seconds per frame, and converged within 150 iterations. This method was able to segment regions where the plume had become very thin that background subtraction alone was not able to recognize. The method was initialized with regions of concentrated gas picked up by background subtraction and filtered to remove noise. See figure **??** for the disturbance map, initialization, and results images.

6. Conclusion

This project dealt with the detection, identification, and segmentation of gas plumes in long wavelength infrared hyperspectral data. Numerous methods of detection and segmentation were applied, as well as a way to visualize the hyperspectral data.

Hyperspectral data consists of wavelength channels for each pixel, transforming simple image processing algorithms into difficult tasks, having to deal with those channels. Our data consists pixels that are represented with 129 channels, whereas a normal image has 3 channels for RGB. Since there currently does not exist a way to visual data in $\mathbb{R}^{129}$, we used PCA to project our data in 3 channels so that we can obtain a false color movie. Due to flickering in the movie, we needed to apply the Midway Equalization method to normalize the frames. With the removal of flickering, we were able to obtain a very nice visualization of the data.

In image processing, background subtraction is a very simple method of preprocessing. Taking two consecutive frames and obtaining the difference between the two, allows for a disturbance map to be created, identifying areas that are moving. We found that applying a weighted running average allowed for the best extraction of moving features. From here, the disturbance map could be fed into semi-supervised methods of detection or segmentation.

Our methods of detection required construction of dictionaries that contained background signatures and target signatures. The methods for background extractions were PCA or ATGP, while we were given a dictionary of target signatures. From here, the dictionaries were using in our detection methods, one a statistical model and the other a method of unmixing. The statistical model was one that attempted to determine the probability that a pixel was either one of the background signatures or one of the target signatures. Unmixing is the method of trying to determine which signatures are present in each pixel. The L_1 method of unmixing utilizes the dictionaries to try to determine the amount of each element exists in the given pixel. We would then see how much of the target gas is present and use that as a method of detection.

Our last work was done in segmentation. We attempted four different methods of segmentation, K-means, spectral clustering, Nyström method, and the Ginzberg-Landau minimization. With the use of K-means, we were

able to determine that the cosine distance metric works best with the data as well as a peculiarity, outlier clusters. Unfortunately, the gas plume was not able to be individually segmented using k-means. Spectral clustering was attempted next, a modern algorithm that utilizes the eigenvalues and eigenvectors to cluster the data into seperate groups. The results of spectral clustering were quite promising, being able to isolate the gas plume from the background signatures. The problem with this method is that it requires user input to select the gas plume and that it is quite time consuming. An alternative to full spectral clustering is the Nyström method. It utilizes a random selection of points to approximate the eigenvalues and eigenvectors. It is not able to form as many clusters as the spectral clustering, nor is it able to isolate the gas, however it is many times faster. The results show that while not sufficient for a stand alone segmentation method, Nyström is good input for other methods, such as the Ginzberg-Landau minimization. The Ginzberg-Landau minimization was the last method of segmentation that we performed. It is able to isolate the gas plume, with the utilization of the disturbance map found from background subtraction.

References

[1] An mbo scheme on graph segmentation and image processing.

[2] Andrea L. Bertozzi and Arjuna Flenner. Diffuse interface models on graphs for classification of high dimensional data. To be published in a 2012 Multiscale Modeling and Simulation, 2012.

[3] J. B. Broadwater, D. Limsui, and A. K. Carr. A primer for chemical plume detection using lwir sensors. Technical report, National Security Technology Department, 2011.

[4] Julie Delon. Midway image equalization. *Journal of Mathematical Imaging and Vision*, 21:119–134, 2004.

[5] Zhaohui Guo, Stanley Osher, and Aurther Szlam. A split bregman method for non-negative sparsity penalized least squares with applications to hyperspectral demixing. *ICIP*, pages 1917–1920, 2010.

[6] Dimitris Manolakis, Christina Siracusa, and Gary Shaw. Adaptive matched subspace detectors for hyperspectral imaging applications. Technical report, MIT Lincoln Laboratory, 2001.

[7] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 14, 2002.

[8] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:1–32, 2007.