Developing Machine-Learning and Image Processing Protocols for X-Ray Image Analysis

Gabriela Basel, Mike Gu, and Kevin Slater,
Internal Mentor: Xiaoying Liu,
External Mentors: Jin Wang,* Nicola Ferrier,† and Wei Jiang ‡

(Dated: March 2019)

Fuel injection is a common method of fuel introduction in a variety of combustion engines – particularly diesel engines, which have become increasingly popular in recent years. The design of fuel injector nozzles is an area of active research. For efficient combustion, fuel spray ejected from the nozzle must quickly break up and mix in the combustion chamber. Understanding patterns of fuel spray breakup is therefore important to design better injector nozzles and, consequently, more efficient engines. Because injection is such a quick process and because fuel spray is optically dense, it is best visualized with high-speed x-ray imaging. In this paper, we introduce protocols to quantitatively analyze fuel spray breakup patterns from x-ray images. In particular, we present methods we developed for x-ray image normalization and segmentation in order to identify individual fuel droplets from fuel spray images. Additionally, we propose methods to cluster and classify similar fuel droplets in order to quantify spray characteristics and compare different types of sprays. Overall, we found these methods show promise in identifying patterns of spray breakup and distinguishing between sprays of different pressures, but our methods require much more data and testing to produce meaningful physical results. Finally, we developed and tested machine learning algorithms to categorize fuel droplets based on shape, which can potentially be used to extend our methodology to new images and large datasets with relatively little computational exertion.

I. INTRODUCTION

Fuel injection – used primarily in diesel engines – has become one of the most common methods of fuel introduction in internal combustion engines, and the increasing popularity of diesel engines has accelerated fuel injection spray research. Diesel engines saw a huge increase in worldwide usage during the twentieth century. However, diesel engines require heavy hardware, and diesel is taxed more heavily than gasoline in the United States, so gasoline remains the fuel-of-choice for the typical United States passenger vehicle. However, diesel engine usage has only continued to rise around the world, finding use in passenger vehicles in many countries, heavy vehicles such as trucks and trains, and non-transport applications such as generator power sources and irrigation pumps. Diesel engines have a number of advantages over gasoline engines which may account for their rising popularity. For example, they have a higher fuel efficiency, and biofuels can be used in existing diesel engines without adaptation.

Fuel-injecting engines such as diesel engines operate through the injection of liquid fuel into an air- and exhaust-filled combustion chamber. A chemical reaction occurs at the interface where fuel and air meet, and the high pressure of the air cylinder causes spontaneous ignition of the fuel upon injection. Because of this, the fuel has little time to diffuse into the cylinder before reaction and ignition. The reaction rate and amount of heat released upon ignition are determined by the rate at which fuel and air diffuse due to injection[1]. This diffusivity due to injection is incredibly difficult to categorize. The fuel sprays released by these injection mechanisms are intrinsically stochastic because atomization results from instabilities of liquid columns and because turbulence/cavitation is prevalent within the injector and near the nozzle due to high pressure of the injection fluid. What's more, the ambient fluid in the release cylinder is often turbulent and is made more turbulent upon injection. Physically complex phenomena also add complications – high droplet number density and significant droplet interaction through collisions and coalescence upon atomization drastically reduce the accuracy of theoretical assumptions [2].

Clearly, this myriad of complications makes high-pressure fuel injection difficult to model or characterize. And, since fuel injection efficacy is heavily dependent on the behavior of the injected fluid and characteristics of atomization, this is a barrier which many have worked intensively to overcome. Currently, internal combustion engine design and fuel development are hardware-intensive and experience based, but – to improve fuel development – we want to transition to simulation-intensive, science-based study. The ultimate goal is not only to be able to prescribe spray characteristics required to produce desired fuel injection results,

* Advanced Photon Source, Argonne National Laboratory, Lemont, IL 60439, USA
† Mathematics and Computer Science, Argonne National Laboratory, Lemont, IL 60439, USA
‡ Argonne Leadership Computing Facility, Argonne National Laboratory, Lemont, IL 60439, USA

but to be able to design a priori spray equipment to achieve these characteristics [2]. We are at a crux in fuel design: in the process of phasing out a rapidly-depleting and ever-tumultuous petroleum market in favor of greener fuel sources. Diesel engines provide a promising opportunity for this smooth transition, because biodiesel fuels can be produced and used in existing diesel engines more readily than in their gasoline analogs. However, a key to successfully generating cleaner and more efficient high-speed fuel injection into combustion chambers is a full understanding of the breakup and atomization mechanism of the fuel sprays [3].

On top of the intrinsic properties of high-pressure fuel injection that make the spray difficult to characterize, high density and complex fluid properties of the spray make the behavior of the fuel injections difficult to *measure*. Conventional techniques have proved unsuccessful at elucidating the internal structure of these high-speed jets because of the multiple scattering by droplets at interfaces and the high density of the jet near the nozzle exit. Focused x-ray beam absorption measurements provide only average quantitative density distributions from repeated imaging. Jin Wang and colleagues at the Advanced Photon Source at Argonne National Laboratory have developed a novel approach to spray imaging based in ultrafast synchrotron x-ray full-field phase-contrast imaging. This technique is able to provide insight into the internal structure of optically dense sprays and instantaneous velocity profiles of the injections. This technique can therefore be used to study transient phenomenon dynamics more closely than ever before possible [3].

Now that this method is available to visualize flow features, efforts have been focused on gathering meaningful information from these x-ray images. Jin Wang and colleagues have developed autocorrelation functions to calculate velocity maps of these sprays [4]. While these functions have produced meaningful results, they require critical assumptions. Namely, these functions assume that the sprays it analyzes demonstrate the feature size and velocity distributions of general hole-type nozzle sprays in which the feature size and velocity decrease with an increase in transverse distance. Our goal is to introduce a workflow which qualitatively analyzes the raw x-ray images in order to elucidate novel features of flow breakup, not based on existing flow or atomization theory, but rather on empirical x-ray image analysis. The large amount of raw x-ray data available makes this problem an ideal candidate for machine learning tools such as artificial neural networks and random decision forests. Given an unknown set of data, these methods sort the data into known categories, which means our qualitative analysis could potentially be extended to analyze novel sprays and injector nozzle designs. Neural networks have long been utilized as highly reliable image classification mechanisms, which leads us to believe that our image processing and machine learning methods lend themselves well to this complex flow analysis problem [5]. Eventually, the type of analysis we present here may lay the groundwork for accurate multiphase flow simulation techniques and provide new insight into flow breakup behavior that cannot be described theoretically.

In this paper, we outline a proposed workflow for fuel spray x-ray image analysis and interpret some of its early results. We have refined our workflow over the course of about five months, but our methods still have a long ways to go before they can be used reliably to produce truly meaningful analysis of x-ray images of fuel injection sprays. Our primary approach was to isolate the fuel droplets present in each image using a variety of image processing technique, and then classify these shapes based on physical properties in order to conduct and translate analysis on the fuel droplet shapes into analysis of the spray as a whole. We provide detailed insight into our process of the past five months, including different methods we tried, challenges we faced, and future directions we find promising. We first provide a detailed analysis of our image processing methods: image normalization, fuel droplet isolation, and fuel droplet shape classification. We then present some analyses we conducted on these droplet shapes and possible insights these individual droplets given into the larger spray. We also provide some machine learning techniques we briefly explored as potential means of expanding our analysis to future images. Finally, we end with a summary of each of these avenues we explored, our workflow which produced the most promising results, and our intuition into the best future directions to take on this project. We hope our results will help inform future research and highlight interesting questions yet to be answered.

II. IMAGE PROCESSING

A. Overview

The sprays we analyze were produced by fuel injections into an ambient fluid in a controlled environment. The x-ray is set up parallel to the spray, such that the images collected contain a full, 2-dimensional picture of the spray with the nozzle at the top of the image and the portion of the spray furthest from the nozzle at the bottom of the image. The fuel is injected into the ambient fluid for a brief amount of time – on the order of 100s of milliseconds. The ultrafast x-ray imaging technology available at the Advanced Photon Source allows the capture of hundreds of frames of the injection process. A single "spray", in this paper, refers to a series of x-ray images taken of a single injection, typically about 130 frames. In addition to images of the fuel injection spray itself, we also have access to x-ray images taken of the ambient fluid both before and after the spray, which can be used for background calculations.

Given a set of raw x-ray images of fuel injection sprays, our goal is to detect fuel droplets present in the sprays and isolate them for comparison. The density of the fuel in a spray is related to the pixel values of its x-ray image by Beer's law:

$$log_{10}\frac{I_o}{I} = A \tag{1}$$

where $\frac{I_o}{I}$ is the transmittance of the sample and A is the absorbance by the x-ray (pixel value). Although the x-ray detector, throughout the course of a spray, absorbs a highly constant amount of light, the x-ray machinery creates artifact fluctuations which change the average pixel value of the x-ray image both through time and through space. These artifacts means that careful normalization is required across all images, because we want to compare fuel droplets across all images, which requires that the background (I_o, in Beer's Law) be constant across all images. The fact that the relationship between transmittance and absorbance is logarithmic means that any scaling we perform during normalization must be divisions across all images, as opposed to subtractions across images as in standard background subtraction procedures. After this careful normalization to ensure a common background across all images, we then rescale the images from raw x-ray absorbance to 8-bit greyscale. After normalization, we isolate the darkest areas of the images (representing the fuel droplets – regions of greatest fuel density where little light could pass through to reach the detector) by removing all regions of the image above a certain greyscale threshold. Once these shapes are isolated, we can then classify and analyze them.

B. Normalization

The first step in image normalization is to remove constant background noise (an artifact of the experimental apparatus and x-ray machine). We are able to approximate the constant background noise by taking an average over the x-ray images of the ambient fluid before the spray begins, which serves as a background to the fuel spray. Figure 1 shows this average background as measured by the x-ray detector. We can see in this figure that there are dark regions caused by constant background noise that may be mistaken for fuel droplets if not removed. To remove this noise, we divide every image by this average background shown in figure 1 (see Equation 1). Note that, in order to construct a large library of labelled fuel droplets, we need to consider several sprays. Because the images of these sprays may be collected under different conditions, it is necessary that we calculate and remove the constant background noise individually for each spray. This method we use allows our protocol to be extended to future sprays and images by normalizing the background of every image to 1.

FIG. 1. Averaged constant background noise.

In addition to experimental noise which is constant across all images in a spray, there are background fluctuations which occur between different images of the same spray, on a timescale much smaller than that of a single injection spray, as a result of changes in environment and imperfections in the x-ray detector. This is illustrated in figure 2 **A**. In this image, each line shows the distribution of pixel values within a single image of a spray. The fluctuations are indicated by the shift in mean between each distribution. Given the narrowness of the distributions in comparison to the fluctuation size, these fluctuations can have a very significant effect on analysis.

We consider two methods of background rescaling in order to minimize the effect of these fluctuations.

Single-Reference-Point Rescaling

The first method, single-reference-point rescaling, uses the image background as its single reference point. This method makes the assumption that the background (the color of the ambient fluid as measured by the x-ray detector) is spatially constant across an image. Upon visualizing figure 3, we can see that this assumption is sufficiently accurate for the images used in our analysis (but note that this assumption may not be so accurate for other data sets). After dividing all images in a spray by the constant background (figure 1), we then divide each image by a single background value in order to make the background value of each image equal to 1, essentially eliminating all temporal fluctuations. This deviation of each image's background from 1 is calculated by taking the average value of a portion of each image – a portion of the image in which no fuel ever reaches throughout the duration of the spray (e.g. in the top corner of the image, because the spray is always condensed near the nozzle at the top center of the image). This value is then divided into the entirety of the image, and this process is repeated separately for each image. This method nearly eliminates temporal

fluctuations, as seen in figure 2 **B**, where the pixel value distribution of each image is very similar.

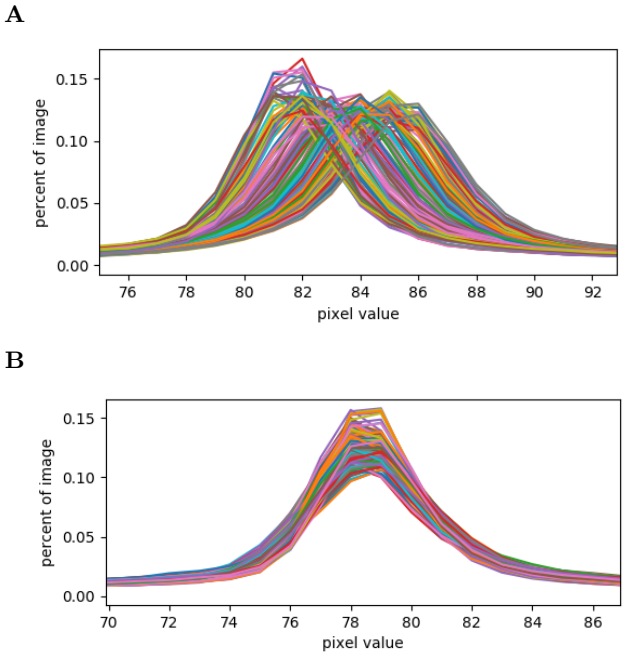


FIG. 2. Image pixel value distributions before (**A**) and after (**B**) single-reference-point rescaling. Each color represents a different image. After rescaling, background fluctuations between images are reduced and the average color of each image is nearly the same. Note that the scale shown on the x-axis is after 8-bit rescaling.

Double-Reference-Point Rescaling

Although temporal-spatial fluctuations (fluctuations which occur on both a small timescale and a small spatial scale) were not significant in the images we analyzed, they may still occur in other datasets. Since these fluctuations occur in both space *and* time, they cannot be captured with just spatial rescaling (figure 1 or temporal rescaling (figure 2 **B**) alone. These types of fluctuations introduce significant analysis issues, but a double-reference-point rescaling method can alleviate some of theses complications. This method takes advantage of the fuel injection nozzle as a constant portion of the image. Because the nozzle, like the ambient fluid/background, is constant throughout the spray, we take its average value as a second reference point. Instead of dividing each image by the average background value as in our single-reference-point method described above, we scale each image using equation 2. Min and Max are the average values of the nozzle and the background in each frame, respectively, and $newMin$ and $newMax$ are c and 1, respectively, where c is a constant value to which we scale the nozzle in every image (we recommend setting c to be the average value of the nozzle across all images prior to double-reference-point rescaling). This will minimize some minor temporal-spatial fluctuations, but will not scale images with serious spatial-temporal fluctuations to be nearly as homogeneous as that in figure 3.

Finally, after all images are normalized, we rescale each image to 8-bit integers $(0-255)$ so they can be read and visualized with a standard computer image visualization software. Our normalization method results in an image comprised of float pixel values, which any image display method will necessarily truncate to integers. From our analyses, we are confident that these truncations do not significantly affect analysis in any observable way, and any effort to minimize the truncations will result in larger file sizes and will needlessly increase computation time.

To rescale our images to 8-bit matrices, we simply compress their current range of pixel values to the new desired range of $0-255$ with the below equation:

$$I_N = (I - Min)\,\frac{newMax - newMin}{Max - Min} + newMin \quad (2)$$

Here, I_N is the matrix representing the new, 8-bit image, and I is the matrix representing the original image. For a color range of $0-255$, we set $newMax = 255$ and $newMin = 0$. Max and Min are set to be the maximum and minimum pixel values over *all* the images that we are rescaling in order to preserve the constant pixel distribution mean across images as seen in figure 2. This method does, however, require saving the values of Max and Min to use in rescaling of future images.

We found that it is common for outlier pixel values to significantly compress the standard deviation of the distribution of pixel values. This can be visualized by the distributions in figure 2 which necessarily have ranges near 255, but which have standard deviations of less than 5. This results in an 8-bit greyscale image dominated by a few grey values, with very few white or black regions (see figure 3). By setting an upper and lower bound on Max and Min in equation 2, it is possible to expand this small, highly prevalent set of greyscale values to take up more of 8-bit space. This allows analysis to more easily distinguish between values in 8-bit space and thus more easily isolate the dark fuel droplets. Since our analysis neglects all pixels above a certain value (we only analyze dark regions which represent condensed fuel droplets), we recommend setting an upper bound on the pre-scaled image (setting Max in equation 2 to a value *lower* than the true maximum pixel value of all images). This does, however, require brute-force trial-and-error calculations with equation 2 to ensure that no important features of the image are lost. This method also requires saving this chosen Max value for use on future images.

Figure 3 shows the end result of the normalization and rescaling process.

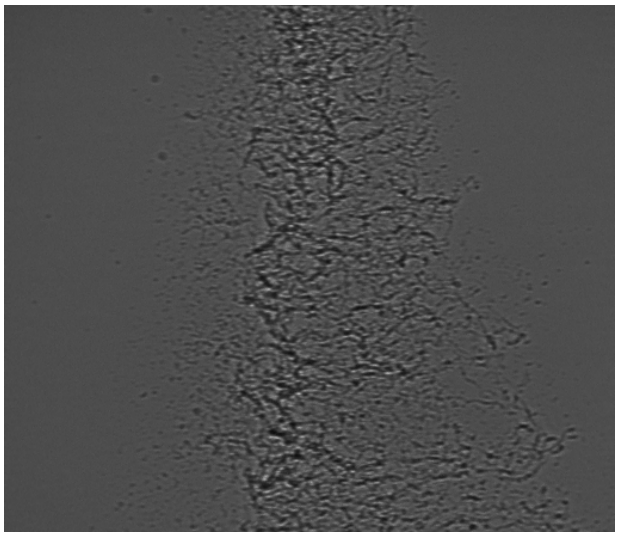


FIG. 3. Frame of an 80 bar spray after normalization and rescaling.

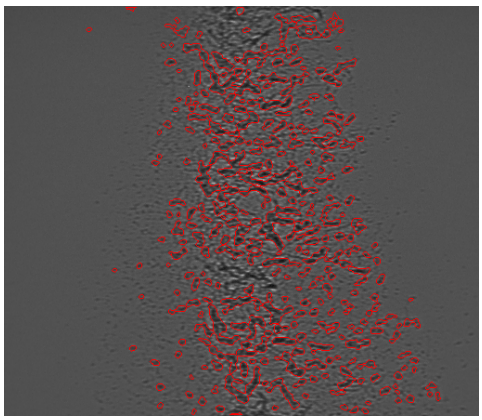
C. Image Thresholding

The normalization steps taken above allow us to use a single process to isolate fuel droplets in all images. To locate the fuel droplets, we exploit the fact that the fuel droplets appear as dark regions in the image, because less light is able to pass through the dense fuel droplets and reach the x-ray detector than other regions of the image (see figure 3). The first significant complication of droplet isolation is the fact that the fuel droplets, though all darker than the background of the image, show a range of greyscale values. This is a result of multiphase flow, where some fuel regions are much denser than others, and small fuel droplets, which necessarily have small volume and therefore provide little obstruction to the x-rays, appear only marginally darker than the surrounding ambient fluid. This can be visualized in figure 3 and quantified by the smooth distributions of figure 2.

Single-Value Thresholding

One method we apply to bypass this complication is single-value thresholding across all images followed by further greyscale analysis. This involves choosing a single pixel value, and turning all pixels in every image with a value larger than this specified thresholding value to white. We can then perform analysis on just the dark areas of the spray. There are multiple methods which can be employed to choose this single pixel value for thresholding. The first single-value thresholding method we consider is the Otsu thresholding method [8]. The underlying assumption of this method is that the image set has a bimodal distribution of colors where the background pixels fall into one peak and the foreground pixels (fuel spray) fall into another peak. The Otsu algorithm then chooses a threshold value between the two peaks of the bimodal distribution that minimizes the variance within the foreground as well as the variance within the background. Since, as just described above, our images clearly do not show bimodal distributions of pixel values (see figure 2), this method fails to isolate a significant number of shapes. Figure 4 shows the results of the Otsu method applied to an image in our dataset, where **A** shows lines drawn along the pixel values chosen by the Otsu algorithm and **B** shows the same image, with all pixels with values above this thresholding value shown in white and all above the thesholding value shown in black. Upon close observation of image **A**, we can see that many of the fuel droplets fail to be detected with this method.

A



B

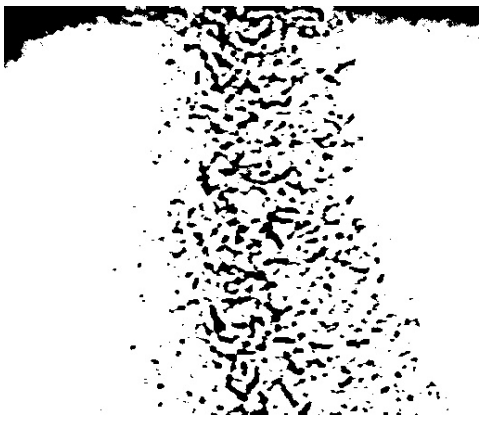


FIG. 4. Result of an image thresholding with the Otsu method. **A** shows the fuel spray with lines drawn on top of pixels with the value chosen by the Otsu algorithm. **B** shows the same image, but all pixels with values above this Otsu thresholding value are turned to white, and all below are turned to black.

In order to improve upon this binodal Otsu method, we employ a slight variation of the Otsu method known as "multi-thresholding," which allows us to assume a different number of peaks in our distribution. With multi-thresholding, we model our single-peak distribu-

tion as a distribution with multiple peaks, and we can choose the number of peaks which produces the best model. We use k-means clustering to find the best peaks (or "centers," in the clustering algorithm) to model our distribution. The k-means clustering method is described in more depth in the next section. Right now, we point out that the primary distinction from Otsu thresholding is that the Otsu algorithm minimizes the variance of each peak while k-means minimizes the total sum squared distance of each point from its nearest peak. We tested multiple different k values (peaks in our model), and then thresholded our images using the second-highest center value returned by the k-means algorithm (a standard practice of multi-thresholding). Figure 5 **B** shows the result of this method for three different k values. This method of shape isolation provides no quantitative metric by which we can distinguish a "good" thresholding value from a "bad" thresholding value. Qualitatively, we want as much of the spray to fall below our thresholding value as possible (requiring a high thresholding value), but we also want the droplets of the fuel spray to remain discrete (requiring a low thresholding value). By observing figure 5 **B**, we observe that a value of k = 3 produces the best balance between these two criteria. Additionally, figure 5 **A** shows an "elbow" at a value of k = 3. This elbow is simply a point in a plot of k versus the total sum squared distance of each pixel value from the nearest clustering center where the decrease in total sum squared distance with higher k becomes too large a sacrifice (as observed in the k = 4 picture of 5 **B** where little of the spray is captured by the chosen thresholding value). For our finalized workflow, we use k-means clustering to cluster the pixel values of all images into 3 clusters, and then we threshold the images on the second-highest k-means center value.

Although this thresholding method works well for our analysis, we still run into many complications later in our workflow, and we acknowledge that this method of choosing a thresholding value is highly qualitative and subjective, and we recognize that this portion of our workflow would benefit from more troubleshooting and different methods.

Now that we have isolated our fuel spray using a reasonable thresholding value, we need to analyze the shapes we have isolated. Figure 6 **A** shows an enlarged portion of the center (k = 3) image from figure 5 **B** so that we can see the individual fuel droplets isolated with our chosen thresholding value. While some shapes are clearly individual fuel droplets, other shapes appear to be large complexes formed by many smaller droplets (a property of multiphase flow). Deciding how to deal with these large fuel complexes is one of the most significant challenges we faced on this project. In some of our analysis, we analyzed these large complexes as if they are individual, large fuel droplets. Here, we present a method for treating these complexes rather as a conglomeration of multiple small droplets.

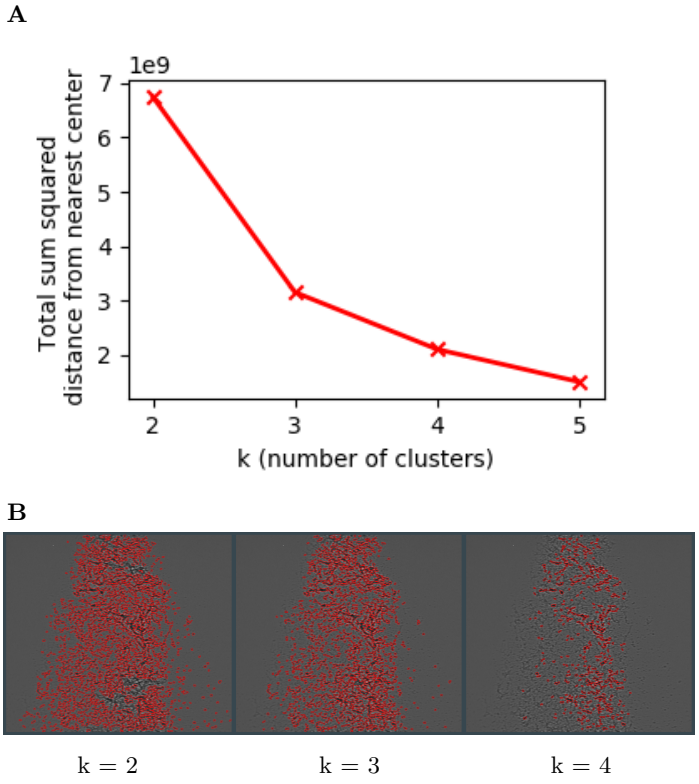


FIG. 5. Using k-means to determine a single value for thresholding. **A** shows the sum of squared distances of each pixel in an image from its closest k-means cluster center, using different k values for clustering. **B** shows the result of shape isolation using the second highest thresholding value obtained from k-means clustering, using different k values for clustering.

Adaptive Thresholding

To separate out the smaller droplets that make up these complexes, we apply adaptive thresholding. A representative case of this method is shown in figure 6 **B**. In this method, we take each large complex and analyze it individually. With each complex, we apply a variety of new thresholding values to the complex and take note of the number of individual contours which each threshold value separates out in the complex. Shape a) in figure 6 **B** shows the original large complex isolated in this case, and all other images show this same complex with new thresholding values applied. Shape b) in figure 6 **B** shows the shape with the thresholding value which isolates the largest number of small shapes within the complex, in theory elucidating the smaller fuel droplets which make up the large complex. In our future analysis, we analyze these small shapes as shown in shape b) and ignore their existence in the larger complex.

Adaptive thresholding (or another method used to isolate small shapes from a larger complex) has the

benefit of reducing an entire image into similarly-sized small shapes, making the flow easier to analyze and model. However, these methods discard the information provided by the spaces between these small shapes which bring the shapes together into a larger complex – a unique property of multiphase flow. For the rest of the analysis we present here, we do not apply methods such as adaptive thresholding beyond our initial single-value thresholding described previously, in favor of keeping the information retained in these large shapes. However, the great heterogeneity inherent to these large complexes necessitates a large amount of data to better characterize them *as* large complexes, which we did not have access to. We later provide some preliminary analysis on these large shapes, but recognize that significantly more data and method testing is required to produce meaningful results.

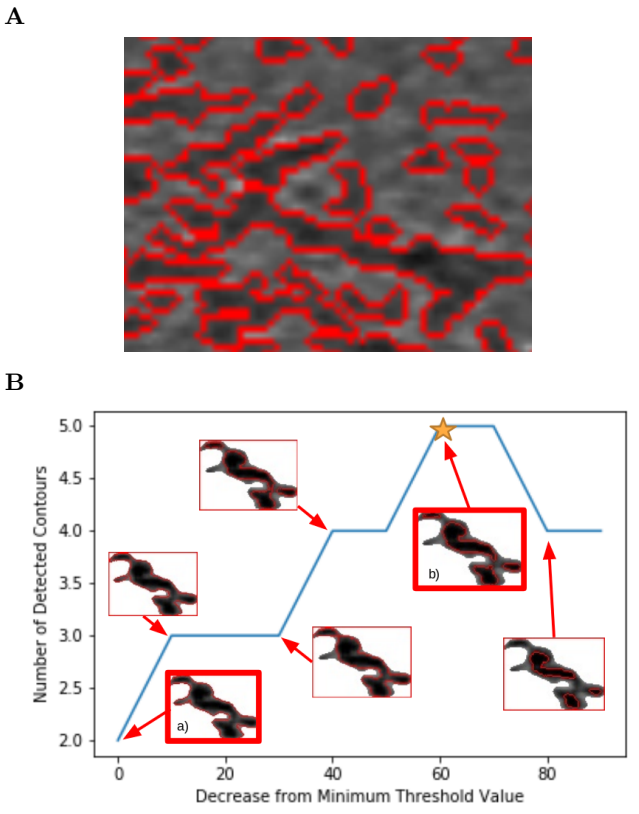


FIG. 6. Adaptive thresholding. **A** shows a portion of an image from figure 5 **B** (k = 3), enlarged to see individual shapes fuel droplets. **B** shows the process of adaptive thresholding applied to an arbitrary large complex. See text for details.

Multiple-Value Thresholding

All of the image thresholding previously described requires first applying a single thresholding value to all images. Another option is to instead analyze each shape individually, regardless of an initial threshold. The most common approach to this option is to treat the image as a topographical map, where darker regions are peaks and lighter regions are valleys. These methods, collectively known as watershed thresholding, typically define shape boundaries along regions of steepest gradients [9]. In theory, this method defines the boundary around the edge of each fuel droplet, regardless of absolute pixel value, because, no matter the density of the fuel droplet, each droplet will have a boundary with a steep gradient. This avoids the problem faced by single-value thresholding methods of defining shapes at only a single density value. However, in practice, these watershed thresholding techniques proved unsuccessful, primarily due to x-ray fringe on the edges of droplets that distort the gradient as well as a relatively low spatial resolution, which does not lend itself well to gradient-dependent methods. We mention these techniques because we think that they may be promising under different circumstances or with different types of flows, but we do not pursue these methods further during our analysis.

D. Shape Isolation

After thresholding our images, we must separate the thresholded regions into individual shapes for classification and analysis.

First, in order to accurately isolate the shapes, we binarize each image to black and white. Regardless of the thresholding method we choose from the previous section, we turn all pixels with values above our given threshold (or all pixels outside of our boundaries drawn along gradients, in the case of multiple-value thresholding) to white and all those below our given threshold to black (see figure 4 **B**). Next, we simply need to separate the droplets into individual shapes. To do this, we define the fuel droplets as contours using the *findContours* function in the openCV Python package [6, 7]. A contour is a continuous curve connecting points of the same color along a boundary – in our case, contours are located around fuel droplets which have been defined using Canny edge detection. The openCV *findContours* package automatically defines and saves a contour with properties such as its perimeter, area, major and minor axes, and location within the larger image. With this shape isolation method, we now have a wealth of data on each fuel droplet in the spray. To construct a library of fuel droplet shapes, we save each contour as its own shape and restore the original greyscale information *inside* the contour, but keep the area outside the contour white (see figure 11 **C** for a visual of these shapes). We are then able to classify the shapes based purely on the droplet (including its density information), disregarding information of its environment.

For the preliminary analysis we present later, shape isolation using *findContours* allows us to neglect some shapes of certain characteristics. For example, in some

analyses, we only want to consider small, single-droplet shapes (ignoring any multiphase flow behavior of the spray). To neglect contours which may be strings of smaller shapes, we can exclude shapes with large perimeter:area or major:minor axis ratios from our analyses. Additionally, we can perform initial shape classification based on parameters such as area or major:minor axis ratios in order to save computational time.

These shape isolation methods also provide a number of promising future avenues of analysis. For example, major and minor axes of a droplet can be use for location droplets which are similar upon rotation around an axis (which is important given the 3-D nature of sprays). Additionally, knowing a contour's nearest neighbors may give insight into how a droplet's shape evolves over time.

E. Shape Classification

After locating a large number of fuel droplets across a variety of different spray images, we want to assign a single label to each of these droplets in an attempt to group together droplets with similar features. This label could be based on any of a droplet's many properties, such as its size, shape, or density. Ideally, each label should be distinctive, and each droplet should be uniquely identified by its label. This simplifies our analysis in three key ways: first, we can classify a spray based on the aggregate qualities of a particular category of droplet. In this way, we can identify and track certain droplet "categories" that may naturally arise during different stages of or locations in the injection spray. Second, we can approximate every droplet within a particular category as being identical. This is a good approximation only for highly distinctive categories, but it can allow us to reduce a set of hundreds of thousands of unique droplets into a much smaller number of categories with known characteristics. Finally, we can use these labels to train a machine learning algorithm to automatically recognize new shapes. As we will discuss later, this has the potential to greatly increase analysis speed and, eventually, compare new sprays to those with known characteristics.

There are several algorithms available for cluster analysis. We chose to implement k-means clustering using the openCV Python package [10]. K-means is a vector quantization algorithm that partitions a dataset into k sets, where k is a number chosen to strike a balance between distinction between different categories and similarity within the same category. The k-means algorithm works by first randomly selecting k points from the dataset (we assign k ahead of time and will talk later about how we determine the value of k). These serve as the initial set of cluster means, or "centers". The algorithm then performs the initial clustering step: for each mean, a cluster is defined to be the set of points closer to that mean than to any other mean. Finally, the algorithm calculates the mean of each of these k clusters, calculated analytically from the set of data points. This process is then repeated with these new means: clustering around them, then re-balancing. This continues until the difference between two consecutive sets of means drops below a given tolerance. K-means is generally considered to be an efficient clustering algorithm and is well-suited for our numerical dataset [11].

The primary downside to this method is that it requires us to pick a number k of clusters. Deciding on a value of k is difficult, in part because there may be no natural division of our dataset into k distinct clusters. In theory, the most accurate number of clusters for N points is N (where each data point has its own mean). However, there are diminishing returns in the descriptive ability of the clusters for each additional cluster we add. Therefore, we aim to find the number k where the total sum of squared distance from each droplet to its nearest center has an inflection point. This is referred to as the "elbow method", and is demonstrated in figure 7 for a subset of detected shapes. However, as is demonstrated in this figure, there often does not exist a clear inflection point. We discuss briefly here and later in our analyses some of the consequences of choosing different k values.

Additionally, note that, to apply k-means clustering to a dataset, each point in the dataset must have equal dimensions. Therefore, before clustering any of our detected droplets, we first pad each droplet image with zeros (extra background) to give them equal dimensions. After each droplet image is padded, we flatten it from a 2-D array of pixel values to a 1-D vector. K-means clustering is then performed on this set of 1D vectors. This method of vector flattening can, especially for small k values, sort shapes that *appear* quite similar by eye into different clusters. However, with larger k values, we do not observe negative effects of this method on shape classification.

By applying k-means clustering to all detected droplets in a large number of images we find that the resulting clusters are determined largely on droplet size (that is, the number of non-zero entries in its flattened vector). Figure 8 **A** shows an image that is part of a dataset that has been clustered with k = 5. Each color shows the boundaries of all shapes belonging to a single cluster (yellow shapes belong to one cluster, red shapes belong to a different cluster, etc). We can immediately see in this image that all small shapes are clustered together (yellow), green shapes are slightly larger, then red, then cyan, and then indigo. The clustering algorithm has essentially classified the shapes into different categories with different sizes and no other distinct qualities. To curb these effects, we find it beneficial to first cluster the shapes manually, without k-means algorithms, by

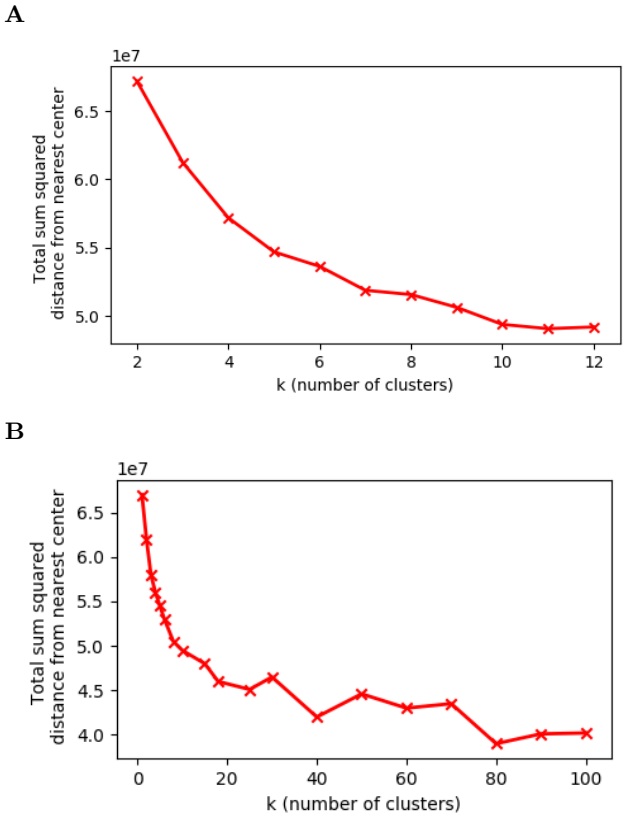


FIG. 7. Using the elbow method to determine the number of clusters to use when clustering small shapes with an area between 10 and 80 pixels. **A** shows identical results to **B**, but enlarged to only show the results at small k values.

size. We perform many analyses, for example, on shapes restricted to an area of between 10 and 80 pixels, which we show in a later section. There are many more small droplets than large droplets present in the sprays we analyze, which lend themselves well to extensive analysis. Large droplets are less prevalent and are inherently more heterogeneous than small clusters, which makes it difficult to create well-defined and distinctive clusters for large droplets (one reason why we consider methods like adaptive thresholding to break up large droplets).

Figure 8 **B** and **C** show the result of this clustering after initial separation by size. **B** shows the same shapes that are highlighted in yellow in **A**, but these shapes have been clustered separately (without considering any other shapes) with k = 4. Note that the shapes are no longer separated only by size, but rather by shape characteristics (shapes in the cluster denoted with orange are more circular, those with green are more elongated, etc.). **C** shows the same shapes highlighted in cyan in **A**, and have now been clustered separately with k = 4. While the results are similar to those seen with the small shapes in **B**, each cluster still remains difficult to distinguish from one another due to the high variation among the different shapes (an inherent property of these large complexes in multiphase flow).

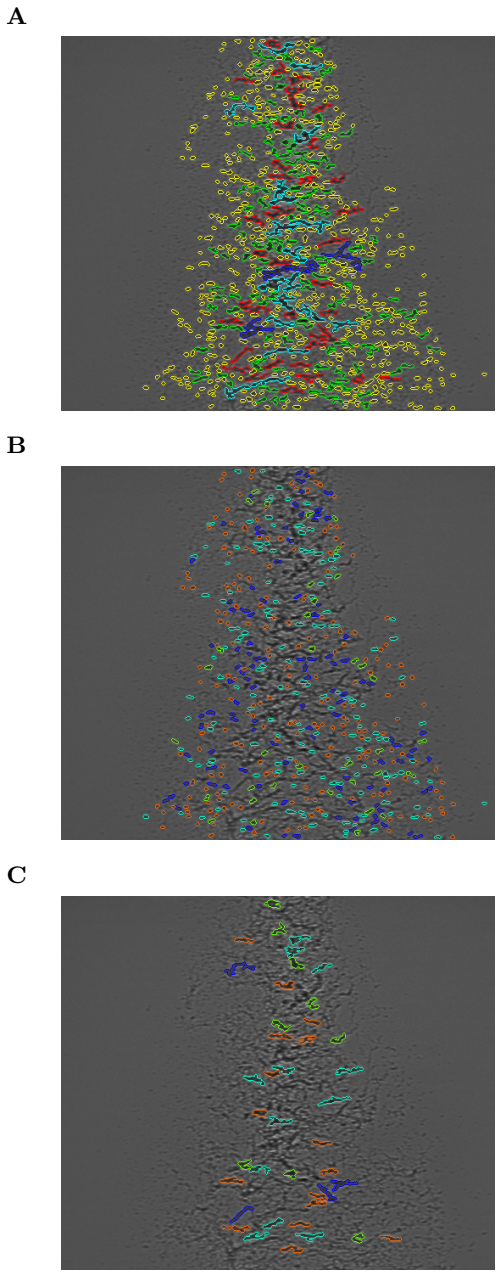


FIG. 8. **A** Representative result of running k-means (k = 5) clustering on all shapes with no prior restrictions. All shapes outlined in yellow belong to one cluster, those in red to another cluster, etc **B** Shows just the small shapes from A (those outlined in yellow), which have been re-clustered with k = 4. **C** shows just the large shapes from A (those outlined in cyan), which have been re-clustered with k = 4.

Because of the large role they play in our analysis, we believe clustering techniques warrant further study. Restricting droplets based on their area (as in figure 8) and height:width ratio prior to clustering may be particularly useful. For example, it may be possible to define

clusters for all small droplets in the spray, then define larger droplets as amalgamations of these known clusters. Future work may consider ways to fine-tune preclustering techniques as much as possible, such as by looking for rotational symmetries in droplets, before using unsupervised learning like k-means to group droplets by characteristics that are more difficult to quantify. Additionally, other unsupervised (or even supervised) learning methods besides k-means clustering may want to be considered, as k-means clustering is very computationally expensive and contains somewhat arbitrary decision-making (like in choosing a k value).

III. ANALYSIS

Ultimately, our goal is to extract meaningful physical results from our image processing analysis. The following analyses are done on four sprays: two at a pressure of 80 bar and two at a pressure of 150 bar. Each spray consists of about 135 frames, for a total of about 540 images used during analysis. Each of the sprays is normalized independently using the normalization process previously described, and then all pixel values of all images in a single spray are shifted such that the mean pixel value of each of the four sprays is equal (using single-reference-point rescaling). Following this normalization process, images from all four sprays are analyzed together. A k value of 3 is used to find a single thresholding value for all of the images, resulting in a thresholding value of 76 (in 8-bit pixel value). Shapes are isolated using binary thresholding and contour detection. These shapes are separated and clustered using a variety of methods (described below) in order to uncover meaningful analysis of the overall spray and flow behavior, based only on parameters of the individual droplet shapes. As previously mentioned, most of our methods depend on a large amount of data, and this macroscopic analysis is no exception. We are confident that some of the trends we explain here are applicable to sprays at the pressures used, but more significant results with physical meaning require more than 540 images.

A. Macroscopic Analysis

One of the most promising methods of extracting such data is by performing macroscopic analyses of the images, using the changing frequencies of the shapes to make conclusions regarding spray behavior. In this section, we show some of the meaningful trends in frequency of different clusters and frequency of different shape sizes both spatially along the spray axis and temporally along the flow axis. We also show how these analyses change when taking into account the area of each shape.

We discussed earlier some of the issues with image classification. Macroscopic analysis depends heavily on the choice of classification parameters. We will show here some of the consequences with various classification and clustering decisions.

First, we present some macroscopic analysis based on broad size parameters. Referring to figures 9 and 10, we can make a few interesting conclusions. For example, from figure 9, we can see that the sprays at 150 bar have more shapes of all sizes than those at 80 bar, suggesting a higher degree of atomization. Interestingly, if we turn to figure 9 B, we can see that the *area* by which the shapes at 150 bar exceed those at 80 bar are nearly equal for all shape sizes (as demonstrated by the distance between the solid lines and the dashed lines). Additionally, as most clearly visible in figure 9, the slopes of each line show little change, suggesting similar breakup patterns along the entirety of the spray axis. In figure 9 we can also see that this is more true at higher pressures, as the slopes of the lines at 80 bar, particularly that for small shapes, does show nonlinear behavior.

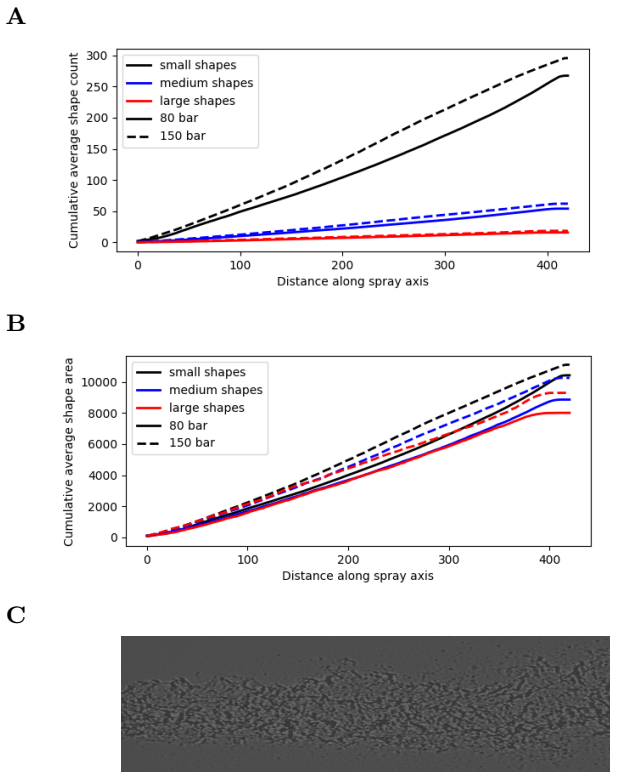


FIG. 9. Small shapes refer to shapes with an area between 10 and 100 pixels. Medium shapes refer to shapes with an area between 100 and 300 pixels. Large shapes refer to shapes with an area between 300 and 1000 pixels. **A** shows the cumulative sums of shape counts along the direction of the spray. **B** shows the same plot, but each shape counted is multiplied by its area. **C** shows the orientation of the spray as measured along the x axis **B** and **C**. Each unit on the x axis refers to one vertical slice of pixels in image **C**.

Turning now to figure 10, we can see that, for most of the spray time, the amount of spray breakup changes little with time and with changes in pressure. However, for all shape sizes and at both pressures, the amount of breakup increases dramatically at the very end of the spray (time greater than 250). This effect is even more significant at a higher pressure, which accounts for the greater number of shapes at higher pressure visible in figure 9. This is an effect we could not see by observing the spray images with the naked eye, therefore providing insight into some of the possible information these shape-based analyses can give.

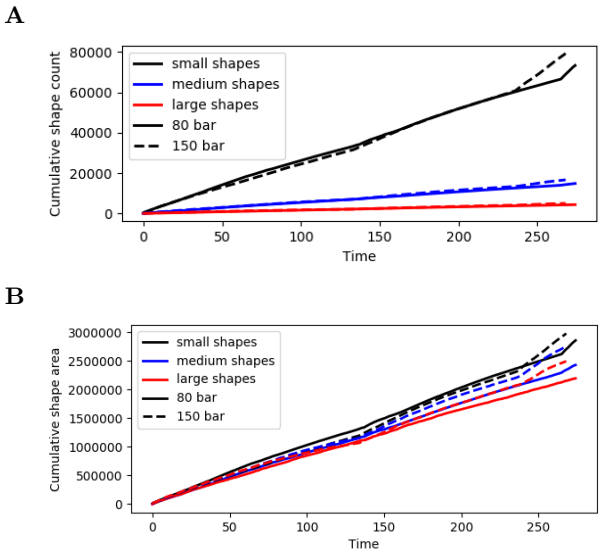


FIG. 10. Small shapes refer to shapes with an area between 10 and 100 pixels. Medium shapes refer to shapes with an area between 100 and 300 pixels. Large shapes refer to shapes with an area between 300 and 1000 pixels. **A** shows the cumulative sums of shape counts through time. **B** shows the same plot, but each shape counted is multiplied by its area. Each unit on the time axis is a different x-ray frame.

Next, we turn our attention to macroscopic analysis using our clustering algorithms. Figures 11 and 12 show the same data as figures 9 and 10, but the data is arranged in clusters computed using k-means clustering, as opposed to being sorted purely by shape area. In figures 11 and 12, only shapes with area between 10 and 80 pixels are clustered, and they are sorted into four clusters (this is the method used in figure 8 B. Figure 11 **C** shows representative shapes of each cluster plotted. From these representative shapes, we can see that the clusters are ordered numerically in ascending size, with clusters 3 and 4 each having shapes of comparable size. By then comparing this information to figure 11 **A** and **B**, we can see that the frequency of the cluster is inversely proportional to the average area of the shapes in each cluster, as seen in **A** (the black line, representing cluster 1 with the smallest shapes, has the largest slope, the blue line, representing cluster 2 with the second smallest shapes, has the second largest slope, etc). The total *area* of the shapes in each cluster present in each figure, as plotted in **B**, is comparable among the different clusters (the black, red, and green lines are all on top of each other). This is true for both pressures tested. This reveals an interesting effect of k-means clustering, which appears to favor clusters of nearly equal total area of the shapes in each cluster, as opposed to an equal *number* of shapes in each cluster.

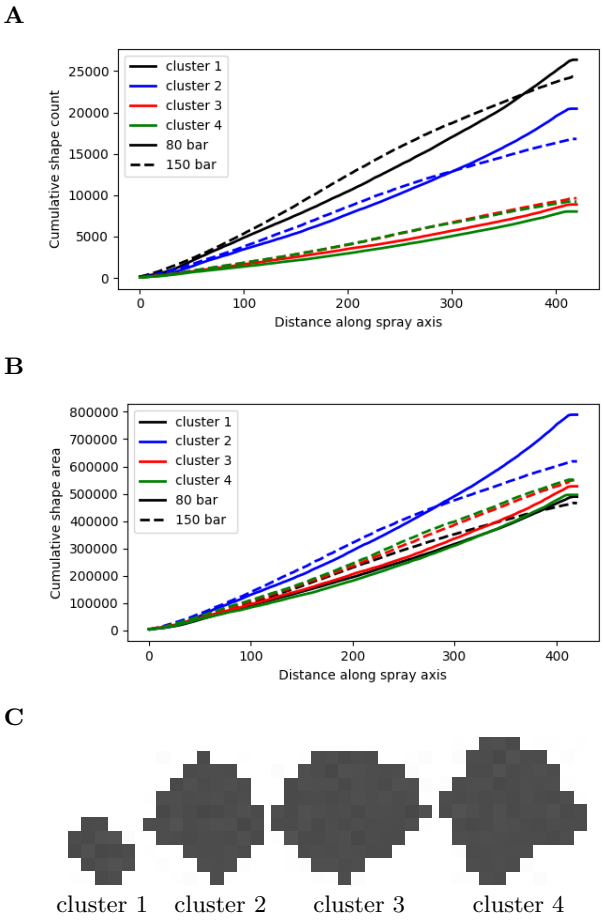


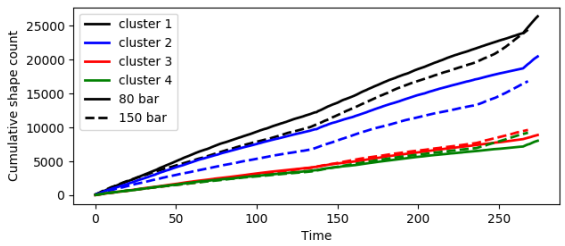
FIG. 11. Macroscopic analysis of sprays at 2 pressures. Only shapes with an area between 10 and 80 pixels are counted, clustered with k-means, k = 4. **C** shows a representative shape from each cluster. **A** shows the cumulative sums of shape counts along the direction of the spray. **B** shows the same plot, but each shape counted is multiplied by its area.

Additionally, a feature we noticed in figure 9 but which is more significant in 11, is the relative linearity of these cumulative shape counts at 150 bar, as indicated by the nearly straight dashed lines in figure 11 **A** and **B**. This indicates that the number of shapes is relatively constant along the spray axis. This, in turn, suggests that, at a pressure of 150 bar, the degree of atomization by the spray is relatively constant along the spray axis. At 80 bar, however, the frequency of shapes increases as we move further along the spray axis, as indicated

by the solid lines of increasing slope in figure 11 **A** and **B**. This behavior is more significant for smaller shapes (ie: clusters 1 and 2), as was also observed in figure 9. This suggests a higher degree of atomization at lower pressures as the spray moves further from the injection site.

Now, we turn our attention to figure 12, which shows the same data as figure 11, but the cumulative counts are sorted temporally as opposed to spatially along the axis. Here, we see very similar data to that in figure 10 – the number of small shapes is nearly constant throughout the lifetime of the spray, with a significant increase at the very end of the spray.

A



B

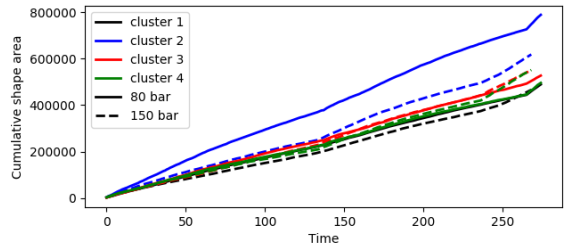


FIG. 12. **A** shows the cumulative sums of shape counts through time. Each unit on the time axis is a different x-ray frame. **B** shows the same plot, but each shape counted is multiplied by its area.

B. Characteristic Cluster Shapes

One of the methods to visualize the effects of our clustering algorithms is to find the "average" shape of each cluster. An average, or representative, shape is found by overlaying all shapes of a given cluster on top of each other. Mathematically, this is done by adding together the image matrix of each shape, where each matrix is padded with white space on all sides in order to give all images the same dimension. This analysis can give insight into the homogeneity of shapes within a cluster, and it can also be used to compare clusters to one another.

Figure 13 **A** shows the representative shape for each cluster of a set of shapes which were clustered with k = 30. The shapes clustered during this analysis all have an area between 10 and 80 pixels (the same shapes analyzed in figures 9 through 12). These representative clusters give a good visual insight into the heterogeneity of the shapes of droplets in the sprays. Even restricting the area of the shapes to within a 70-pixel range, the shapes take on a wide variety of shapes and sizes. On the other hand, the shapes are somewhat homogeneous as well. First, recall that the image of each shape in a given cluster is padded with white space on all sides in order to give all images the same dimensions as those of the largest image in the cluster. With this in mind, the relatively minimal blue area (representing pixels shared by very few of the shapes in a given cluster) surrounding the yellow areas (representing pixels shared by nearly all of the shapes in a given cluster) of all shapes in figure 13 A indicates that there are *no* significant outliers in any of the clusters, and all of the shapes in a given cluster are relatively homogeneous (we will see later that this is less true for a smaller number of clusters). This speaks to both the relative homogeneity of the droplet shapes in a spray as well as to the accuracy of k-means clustering in distinctively classifying the shapes.

One issue with our methods which we can visualize with these representative shapes is the inability to recognize rotational symmetry. As seen in figure 3, the spray axis is vertical. This means that a shape is identical, with respect to the spray, when it is reflected over the y axis. Observing figure 13, we can see that some clusters are, crudely, reflections of each other over this vertical axis. With our current methods, this is not something which we can observe without visualizing the shapes.

Figure 14 shows the same data as figure 13, but the shapes are clustered using k = 4, as opposed to k = 30. With this representative shape visualization method, we can immediately visualize a number of differences between these two clustering results. Since these two figures have the same data, just clustered different, we can first note that all of the shapes represented in the 30 clusters of figure 13 are now contained in the 4 clusters shown in figure 14. We can immediately see this effect in the significant shape heterogeneity contained in each cluster, as indicated by the wide blue areas surrounding each yellow shape in figure 14 **A**.

On the other hand, also notable in figure 14 is the relative symmetry of the representative shapes around the vertical axis. This eliminates the problem present with more clusters, as described above with figure 13, in which shapes which are reflections of each other over the vertical axis are sorted into different clusters.

As described in section II.E. (Shape Classification), the elbow method (a purely mathematical method) is

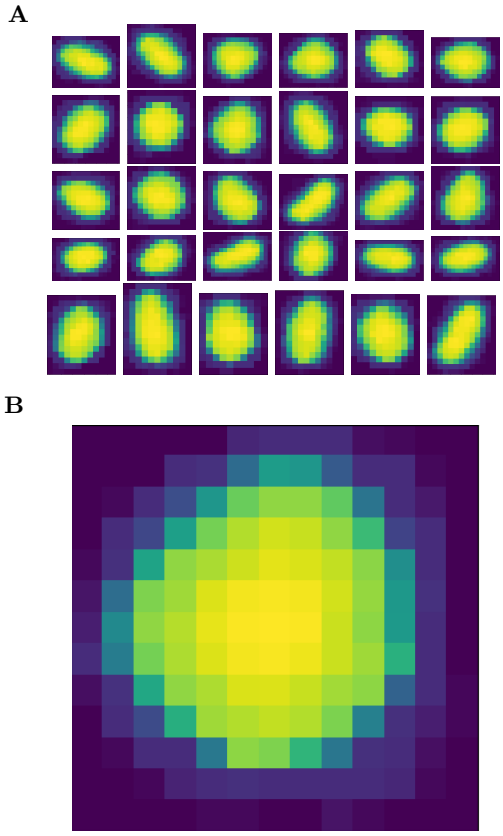


FIG. 13. **A** shows the representative shape of each of 30 clusters. The clusters were calculated using k-means on a group of shapes with area between 10 and 80 pixels. **B** is one of these representative shapes, enlarged to see each individual pixel. Yellow pixels represent pixels which are shared by most of the shapes in a given cluster. Dark blue pixels represent pixels which are only present in a few of the shapes in a given cluster.

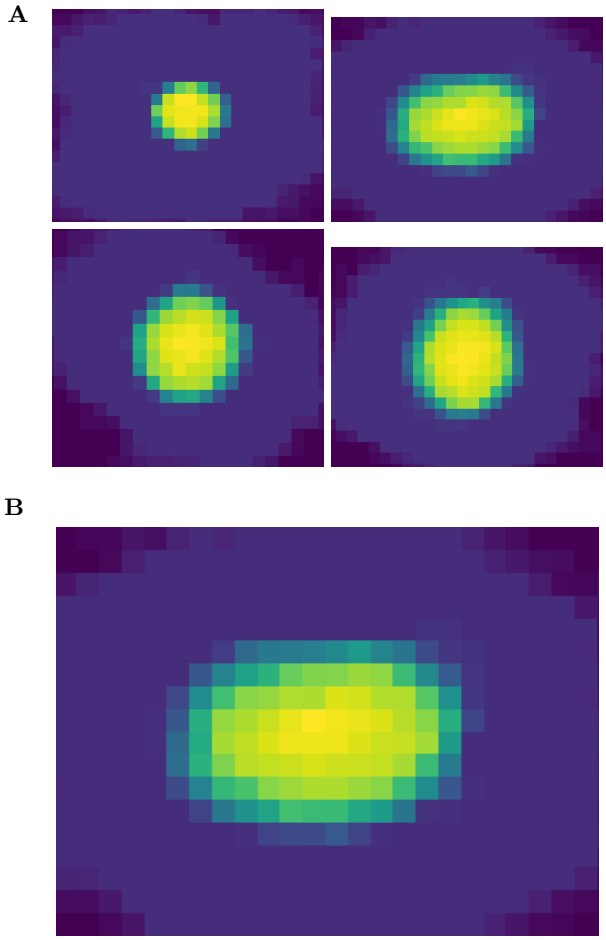


FIG. 14. **A** shows the representative shape of each of 4 clusters. The clusters were calculated using k-means on a group of shapes with area between 10 and 80 pixels. **B** is one of these representative shapes zoomed in to see each individual pixel.

traditionally used to determine the optimal number of clusters to use in order to strike a balance between this homogeneity vs heterogeneity within a single cluster. However, in our case, physical data (such as this vertical symmetry) is more important in determining the number of clusters to use. Although figure 14 shows that, with a small number of clusters, a significant amount of shape information is lost by grouping many heterogeneous shapes together in one cluster, there are many indications that this lost information is not sufficiently relevant. Besides the reflectional symmetry just described, figures 11 and 12 show that the behavior expected from each representative shape's relative size is reproduced by the clusters (small shapes are more prevalent than larger shapes, etc. Additionally, as will be elaborated on in a later section, some machine learning methods have very high accuracy in classifying these shapes into their appropriate clusters with only a small amount of highly heterogeneous clusters.

However, shapes separated with a small amount of clusters lose nearly all information besides relative shape area (as indicated by the fact that the representative shape of each cluster shown in figure 14 is highly symmetrical over both the vertical and horizontal axes). Whether this lost information is truly physically relevant or not is not something which we have been able to deduce from our methods alone and in fact requires true physical information, which will be briefly elaborated on at the end of this report.

IV. SCALING UP

As mentioned previously, k-means clustering is a particularly intensive computational process that can scale up dramatically with large amounts of high dimensional data. For these large amounts of shapes, we look to supervised machine learning algorithms to provide robust shape classification at relatively low computational cost. Additionally, once a significant amount of images

Subcategory	Number of Shapes	Shape Size, x	Number of Clusters
small	112997	$21 < x < 147$	4
medium	19886	$56 < x < 378$	3
large	5407	$132 < x < 925$	4
larger	1292	$299 < x < 1980$	5
largest	135	$675 < x < 4248$	4
30 small	112997	$21 < x < 147$	30

TABLE I. Subcategory information. Each subcategory is a division of a 150,000 shape dataset formed by k-means clustering on shape area. The resulting shapes in each subcategory have a shape area (in pixels) within the bounds given in the third column. All of the shapes in each subcategory have been further clustered using k-means clustering with a k value specified in the fourth column.

have been analyzed, these machine learning methods can potentially be used to apply our shape classification and analysis to new, harder-to-characterize sprays.

To train and test these supervised machine learning algorithms, a large amount of training data is required. To create our training data, we simply run through the steps of our clustering workflow, ie: normalization, thresholding, and k-means clustering, on a large set of x-ray images. We used the Midway supercomputer at the Research Computing Center on the University of Chicago to parallelize this process.

We divided our training data into five subcategories of shapes: 'small', 'medium', 'large', 'larger' and 'largest' (see table I). These subcategories are defined by running k-means clustering on shape area. All of shapes within each of these subcategories are then clustered using k-means clustering on all shapes which fit the given subcategory size restraint. The optimal number of clusters (k value) to be used is determined using the elbow method (see figure 7. More information about these clusters is visualized in table I. Overall, the 'small' subcategory has the most shapes, by far, and the 'largest' subcategory has the fewest. We will later see that these disparate differences in training dataset size have a direct effect on the accuracy of our models.

A number of machine learning algorithms were trained and evaluated on this training data, most notably convolutional neural networks and random forest models. Convolutional neural networks aim to capture the multi-dimensional spacial relationships in images using a series of convolution and pooling layers, which extract features from and down-sample the input images. Random forest models aggregate and average the classification outcomes of numerous decision trees. Gradient boosting, a variant of the ensemble decision tree method, was trained, but was ultimately found to be too computationally inefficient. LightGBM, a recent variant of gradient boosting optimized for speed and accuracy, was also considered, but discarded due to lack of support in our workflow for multi-classification.

Our convolutional neural network was built using three convolution layers and two pooling layers, and optimized with the adam optimization function, coded using the Keras python library. Our random forest models were built with 100 estimators and a max-depth of 50, coded using the scikit-learn python library. 50 was the highest max-depth parameter tuned – higher max-depths could potentially produce even more accurate results.

To train these models, for each size subcategory, we set aside 80 percent of the data for training and 20 percent for testing. We also performed 5-fold cross validation to ensure that our results are not due to a "lucky" specific split of the training data. We end up with several models suited to classifying the clusters within each subcategory, as well as some models that are suited to classifying clusters *across* different subcategories.

Our accuracy results for these algorithms are visualized in table II. Our convolutional neural network underperforms, achieving accuracy results only slightly above guessing. The most compelling possible reason for this unsatisfactory result could be that, because our training labels were determined using k-means clustering on flattened image data, the convolution and pooling layers used on 2-D image data only serve to obfuscate determination of these labels.

Our random forest models, however, are relatively successful. Trained exclusively on the 'small' subcategory, which contains more than 100,000 shapes, our random forest model returns an accuracy of over 98 percent. This means that random forest is able to label the image clusters within the 'small' subcategory with nearly the same results as if applying k-means clustering on all of the data at once, but at a fraction of the computational cost. Random forest performs less well on the other subcategories, however, returning a middling 60 percent on the 'largest' subcategory. This is not surprising, given that the largest subcategory contains only 135 shapes, with little more than 30 shapes for each of its four clusters. This is not enough data for robust classification. As we can see by comparing tables I and II more data results in higher accuracy.

To resolve the issue of a lack of data for certain size subcategories (namely, for large shapes), we can combine subcategories. There are two ways to combine subcategories. One is to preserve the distinct labels for the clusters within each subcategory (i.e. combining one subcategory with four clusters and another subcategory with three clusters results in a new subcategory with

Classification Method	Subcategory(s) trained on	Accuracy
Neural Net	13,000 size subset of shape dataset	0.26
Random Forest	small	0.9876
Random Forest	medium	0.9357
Random Forest	large	0.8076
Random Forest	larger	0.7014
Random Forest	largest	0.5909
Random Forest	distinct combination of large, larger, largest	0.7206
Random Forest	homogenized combination of large, larger, largest (newlarge)	0.9363
Random Forest	distinct combination of small, medium, newlarge	0.9498
Random Forest	homogenized combination of small, medium, newlarge	0.9752
Random Forest	30 small	0.9836

TABLE II. Results of machine learning techniques used to classify shapes. Random Forest models were trained on various subcategories and combinations of subcategories. Distinct combination means that clusters within each subcategory are preserved after combination. Homogenized combination means that clusters within each subcategory are homogenized together into one label after combination. For example, a distinct combination of labels (0,1,2,3) and labels (0,1,2,3) for two different subcategories would result in (0,1,2,3,4,5,6,7). A homogenized combination of these same labels would result in (0,0,0,0,1,1,1,1). Newlarge is merely a homogenized combination of the large, larger, and largest subcategories.

eight clusters). The other is to simply define a single label for each subcategory that is combined (i.e. combining two subcategories creates a new subcategory with two clusters). The latter method essentially removes the different cluster labels within each subcategory, but ensures that each label contains more data. We call the former method a 'distinct combination', and the latter method a 'homogenized combination'. Our problem with lack of data is especially apparent with the 'large', 'larger', and 'largest' subcategories, which contain less than 10 percent of the data altogether. These subcategories also happen to have the lowest accuracy scores. Training a new random forest model on a distinct combination of these subcategories returns an accuracy that amounts to an average of the accuracies of each subcategory, around 72 percent. However, training on a homogenized combination returns a much higher accuracy of 93 percent. This means that the random forest is able to distinguish between the 'large', 'larger', and 'largest' subcategories with 93 percent accuracy. We call this homogenized combination of 'large', 'larger', and 'largest': 'new large'.

We want a random forest model that can distinguish *between* all subcategories, and, even better, a model that can distinguish between all the clusters within subcategories across all subcategories. To accomplish the former, we can train a model on a homogenized combination of all subcategories, and, to accomplish the latter, we can train a distinct combination of all subcategories. However, we have shown that the clusters within the 'large', 'larger', and 'largest' subcategories contain too little data to produce meaningful results. We can train homogenized and distinct combinations of subcategories 'small', 'medium', and 'new large' to remedy this issue. The homogenized combination random forest model returns an accuracy of 98 percent. The distinct combination random forest model returns an accuracy of 95 percent. Although the distinct model is less accurate than the homogenized model, it is arguably more valuable because it is able to distinguish between the clusters within subcategories at only a 3 percent loss. The distinct combination model is our seminal machine learning model contribution for shape classification.

As an additional experiment, we trained a random forest model on 30 clusters within the 'small' subcategory. Because the 'small' subcategory contains close to 70 percent of the data, we wanted to see if a large number of clusters within this subcategory could be classified accurately. The random forest model produced an accuracy of 98 percent, an indication of the volume of the data and perhaps tangible physical trends within the small shapes.

V. SUMMARY OF WORK-FLOW, CHALLENGES AND FUTURE DIRECTIONS

After five months of exploring methodology and collecting data, we have a lot to show, but we want to communicate our biggest takeaways which could potentially inform future analysis.

First, we provide a summary of our work-flow:

- Image normalization
  - Most successful: Dividing all images by constant background computed from an average of the ambient fluid, followed by a division of each image by its average background to bring the background of each image to 1
  - Most significant challenges: Inability to handle images which have high-frequency fluctuations both spatially and temporally, and narrow pixel value distributions caused by pixel outliers
- Image thresholding

  - Most successful: Using multi-thresholding (k-means clustering) to find a single thresholding value to separate the image into dark, fuel-rich regions and light, background regions
  - Other avenues explored: Using adaptive thresholding to handle large complexes representative of multiphase flow, and applying a watershed thresholding method to capture local changes in the greyscale representing finer fuel spray features

- Shape isolation: Defining shapes using openCV Python *findContours* function
- Shape Classification
  - Most successful: Sorting shapes by area, and then using k-means clustering to further separate each area category into a number of categories defined on qualitative measures that can't be assigned manually
  - Most significant challenges: Determining how to define the categories
- Analysis
  - Interesting results: Insight into the degree and type of atomization of sprays at different pressures, both through time and space, and insight into the variance of shapes in multiphase flow
  - Most significant challenges: Translating shape-based analysis into significant physical meaning
- Applying our methods to future images
  - Most successful: Using random forest machine learning methods to assign labels to shapes in new images

We believe that this general workflow can be troubleshooted to produce meaningful results, but, first and foremost, a much larger amount of images must be analyzed to produce meaningful results. Large complexes of shapes are a key component of multiphase flow, but their incredible heterogeneity calls for a large dataset to obtain any meaningful classification of such shapes. Once a sufficient amount of data is processed, one may find that some methods we had previously dismissed may in fact be much more relevant with more data.

For machine learning, a number of steps can be taken to further optimize our models. Concerning neural networks, it might be advisable to test a neural network without convolution or pooling layers on the data, since we are classifying data labeled by k-means on 1-dimensional arrays. For random forest, further hyperparameter tuning on max-depth, number of estimators, and other parameters can potentially yield better accuracy results. It also would be interesting to see the models tested on metrics such as receiving operating characteristic score.

Most importantly, however, we are interested in the physical meaning of our shape classification. A machine learning model for classifying shapes is useless if the labels have no physical meaning. There are obvious unique characteristics of our clusters such as size, but more nuanced macro-scale analysis needs to be done on these clusters. One possible avenue of future analysis is separating clusters by volume as opposed to area, since each droplet exists in 3-D space. Ultimately, we admit that many of our ideas and methods can be significantly improved upon, but we do believe that we have contributed a great deal by rigorously approaching the complex problem of analyzing multiphase flow with purely image-based analysis. We hope that some of what we have learned can eventually be used to inform truly physically meaningful flow analysis.

[1] Manley, Dawn K., Andrew McIlroy, and Craig A. Taatjes. "Research needs for future internal combustion engines." *Phys. Today* 61, no. 11 (2008): 47-52.

[2] Fansler, Todd D., and Scott E. Parrish. "Spray measurement technology: a review." *Measurement Science and Technology* 26, no. 1 (2014): 012002.

[3] Wang, Yujie, Xin Liu, Kyoung-Su Im, Wah-Keat Lee, Jin Wang, Kamel Fezzaa, David LS Hung, and James R. Winkelman. "Ultrafast X-ray study of dense-liquid-jet flow dynamics using structure-tracking velocimetry." *Nature Physics* 4, no. 4 (2008): 305.

[4] Moon, Seoksu, Yuan Gao, Jin Wang, Kamel Fezzaa, and Taku Tsujimura. "Near-field dynamics of high-speed diesel sprays: Effects of orifice inlet geometry and injection pressure." *Fuel* 133 (2014): 299-309.

[5] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." In *Advances in neural information processing systems*, pp. 1097-1105. 2012.

[6] "Structural Analysis and Shape Descriptors". OpenCV.org. *https://docs.opencv.org/3.3.1/* (accessed March 21, 2019).

[7] Suzuki, S. and Abe, K., Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP 30 1, pp 32-46 (1985)

[8] Nobuyuki Otsu (1979). "A threshold selection method from gray-level histograms". IEEE Trans. Sys., Man., Cyber. 9 (1): 6266.

[9] Serge Beucher and Fernand Meyer. The morphological approach to segmentation: the watershed transformation. In Mathematical Morphology in Image Processing (Ed. E. R. Dougherty), pages 433481 (1993).

[10] K-Means Clustering in OpenCV. OpenCV.org. *https://docs.opencv.org/3.0-beta/doc/py_tutorials/* (accessed March 22, 2019).

[11] Jain, Anil K. "Data clustering: 50 years beyond K-means." Pattern recognition letters 31, no. 8 (2010): 651-666.