Cell Detection in Microfluidic Channels

Nicha Apichitsopa Boying Meng Hayley Song
Department of Electrical Engineering and Computer Science, MIT
32 Vassar Street Cambridge, MA 02139
nicha@mit.edu boying@mit.edu hjsong@mit.edu

Abstract

In order to study the properties of cells, microfluidic systems manipulate cells such that the cells with different properties can be distinguished either in their distribution over space or over time. Therefore, the analysis of cell trajectories inside microchannels is a critical part of many microfluidic systems. While Open Source and commercial cell detection and cell tracking softwares are available, these softwares are largely limited to specific applications, e.g. the contrast requirement between cells and background, bright-field versus fluorescent imaging, number of cells that can be tracked, etc. Automated and robust cell detection and tracking algorithms are needed to reliably extract cell positions over time for general applications. In this report, we demonstrate the ability of our machine-learning detection algorithm to detect cells and compare it to the classic image segmentation method. Classic image segmentation method shows robust detection of cells that meet the criteria in terms of size and intensity where machine learning algorithm demonstrates higher flexibility for cell imaging conditions.

1. Introduction

Oftentimes microfluidic systems are designed such that the interested properties of cells are encoded in their positions relative to the reference point or relative to time frames. A key step of this process is; therefore, the detection and labelling of cells versus background and noise. Cell detection is a challenging task the exploration of innovative methods including machine-learning techniques is of interest due to the following reasons. First, cells are heterogeneous. They vary in size, shape, and optical transmittance. The heterogeneity of cell populations are reflected in their image representation. In fluorescent imaging, cells are labelled with fluorophores and the fluorescent emittance significantly helps distinguish between cells and background. However, in bright-field imaging, cells are not labelled and their images exhibit less contrast to the background. Images of cells that are small and highly transmissive are harder to detect because they show less contrast to the background and sometimes have smaller areas than the thermal noise, which renders the classic intensity thresholding and area thresholding method impossible. Second, cells are deformable objects. With pressure from the fluid flow, cells are squeezed through the channels that are narrower than their diameters and can easily change shape from regular circles to distorted circles and ovals, depending their surrounding channels. Lastly, there are some experimental limitations in video recording such as thermal noise and slight changes of illumination or channel positions over time which means that the thresholding parameters need to be tuned differently for fluctuating experimental conditions. Fine-tuning these thresholding parameters can become burdensome for increasing numbers of experiments and recordings. The robustness of machine-learning classification and detection techniques have been shown as they are used to solve several complex problems in image processing such as object recognition, scene recognition, scene analysis. Machine-learning classification and detection techniques may offer solution to the cell detection problems.

2. Related Work

With the advances in optics and imaging systems, there is an increasing interest from biomedical researchers to visualize their experiments by automatically detecting and tracking the cells in a sequence of images. Such automatic process removes the burden of manual inspection and could even detect objects unrecognizable by human eyes. Current state-of-the-art algorithms can be classified into two main categories: model-based approaches and feature-based approaches. Model-based detection algorithms create a model for each object to be detected and update the model as they observe more data. For example, active contour methods such as the one based on the level-sets[6] first represent each object (cell or nucleus) using a separate level-set function. This approach has proven to be suitable for capturing topological chances via its model updates. However, it suffers

from significant drawbacks in computational cost and inability to capture dynamic changes, especially if the object undergoes a significant shape change over a short period of time. These drawbacks limit its applicability to real-world cell imaging systems. Another example of the model-based approach is the coupled mean-shift algorithm [5]. This algorithm requires human operators to select cells to be detected and initialize the object appearance models at the initialization stage. As a video recording of cell experiments often contain hundreds of target objects, this approach is not practical. The main strategy of feature-based approaches is correspondences matching over time, and does not involve any model initialization of update of the target objects. It first segment and locate images and then associate the targets among frames over time. Intensity thresholding [4], gradient detection, and watershed algorithms are some of the well-known segmentation methods. This approach has an advantage in that it does not require any explicit modeling, yet is more vulnerable to image noise, artifacts and intensity variations. In particular, it is still an active research area to find out a robust and efficient way to tune the parameters for the feature-based approach. One approach that has recently been received a lot of attention is to learn from the data via machine learning techniques. Our project takes this approach and focuses on compares various feature extraction methods/filters based on its detection accuracy when combined with a Support Vector Machine (SVM) classifier [7].

3. Approach

For our application, we use a common microchannel structure that replicates the microcapillary network [8]This channel features represents the general microfluidic channels that enforce cell deformation. The murine interleukin-3 dependent pro-B cell line (Ba/F3 cells) is used as the cell sample. The experiment is observed using Zeiss Microscope with 10X objective and is recorded by 12-bit LAVision Imager QE camera with 1280x320 pixels (resolution 1.07 m per pixel) at the frame rate of 20 frames per second. The videos are saved as .mat files. All codes are implemented in MATLAB.

3.1. System Structure

Our system first takes one frame of the recorded video, and crops it into overlapping windows of a fixed size. Each window is then put through a low-pass filter with symmetric padding for noise reduction. From these smoothed windows, a feature extraction technique is used to extract spatial features. The features are concatenated into one single vector per window. The feature vector is used for Support Vector Machine (SVM) binary classification after being standardized by centering and dividing columns by their standard deviations. The SVM model is trained using a limited hand-labeled dataset from pre-recorded videos. Thresholding and non-maxima suppression is then used based on the spatial distribution of the windows marked as containing a cell by SVM to obtain one set of [x,y] coordinate per cell in the original frame of the window. The coordinates of cells are used as inputs to cross-frame tracking algorithms.

3.2. Acquisition of Dataset

The training and validation dataset is acquired in a semi-automated manner. The script is generated such that each frame in the video is shown sequentially and the user can click on multiple locations of cells on each frame. The location of cells are saved as the [x,y] coordinates. Windows of different pixel sizes (16x16, 24x24, 32x32, 40x40, and 48x48) are then placed over each manually-inputted cell location such that the window and the cell share approximately the same centers in order to crop the cell regions and generate the positive training set, which will be called cell windows in this report. The sliding windows of the same sizes with the sliding step of 4 pixels are then placed over the rest of frame (i.e. the regions with no cells) to create the negative training set, which will be called the no-cell windows. Symmetric padding is applied to each frame before cropping process begins.

3.3. Performance Evaluation Scheme

1. Model selection and parameter tuning

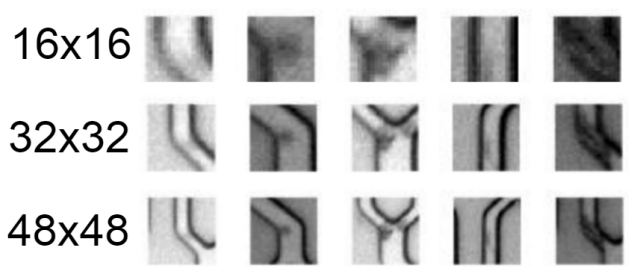
   In order to maximally utilize our limited hand-labeled dataset, we performed cross-validation with the SVM trainer using the dataset. False negative and false positive rates (rFN and rFP, respectively) of labels given by trained SVM classifier to the cropped windows were calculated to compare performances of different feature extraction schemes and determine a subset of usable ones.

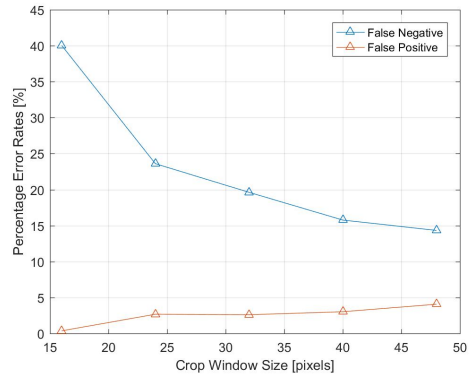
   The cross-validation method was also used to fine tune some of the parameters associated with each feature extraction method, as well as the those used in preprocessing steps.

2. Cell detection and tracking performance

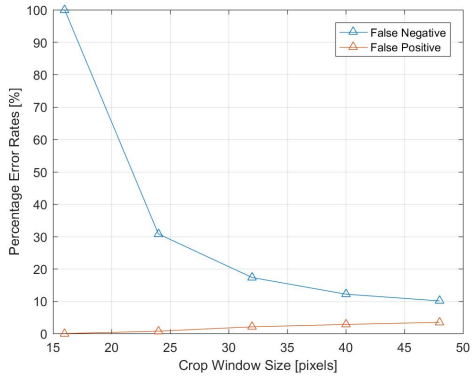
   Quantitatively, the number of cells that are detected, those are missed, and false detections are recorded in a test video from 150 frames of a video recorded in a different experiment than the one from which the training and validation set is acquired. The correction detection (true positive) rate is then calculated as an estimated expectation.

   Qualitatively, we compare the results from background subtraction plus intensity thresholding and our proposed machine-learning approach by visualizing the cell detection and tracking results of both methods in





(a)



(b)

Figure 1: Crop window size study: Percentage error rates as crop window size increases for (a) 4x4 HOG window and for (b) 8x8 HOG window.

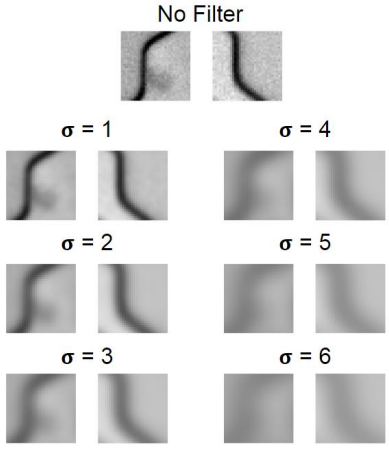
videos using the same input frames. In particular, we look for whether our approach makes improvements in aspects that are shortcomings of the non-machine-learning algorithm.

4. Experimental Results

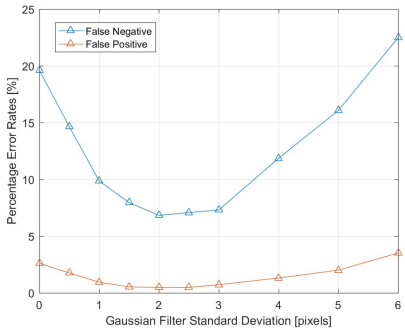
4.1. Pre-Processing

1. Cropping Window Size
   Cropping window size is an important parameter since the cropping window size determines the cell-area-to-background-area ratio. Different cropping window sizes (16x16, 24x24, 32x32, 40x40, and 48x48) are used to create the training set in order to optimize this ratio. Figure 1a and Figure 1b show the inverse relationship between rFN and rFP as crop window size varies for both the 4x4 and 8x8 HOG feature extraction. A crop window with the width of 16 pixels, which is only slightly larger than the cells, is less affected by background channel features than the crop windows of larger widths, but because of the smaller width, the 16x16 crop window is more susceptible to thermal noise in the images.

   Furthermore, number of the extracted features per window increases with the crop window size and can increase the computational complexity of the classifier. In order to balance the rFN and the rFP as well as the computational complexity, we use the 32x32 crop window as the default training set unless specified other-



(a)



(b)

Figure 2: Gaussian filter study: (a) cell and no-cell windows after gaussian filter with different standard deviation is applied. (b) Percentage error rates versus the applied standard deviation

wise.

2. Denoising Filter
   The effects of thermal noise in the images and consequently the extracted features can be reduced by applying the gaussian filter. Gaussian filters with varying standard deviations ($\sigma = 0.5, 1, 1.5, 2, 2.5, 3, 4, 5, 6$) are used separately before each frame is cropped. Note that the Gaussian filter window size is set to be six times the standard deviation and that symmetric padding is used for the image borders. Figure 2 illustrates the rFN and rFP trends as the standard deviation changes. The rFN and rFP are minimal at the standard deviation of 2. As the standard deviation increases from 0 to 2, both rFN and rFP decrease as the high-frequency noise is attenuated. However, as the standard deviation increases from 2 to 6, the rFN and rFP increase as the cell features are also lost or attenuated by the gaussian filter.

4.2. Feature Extraction

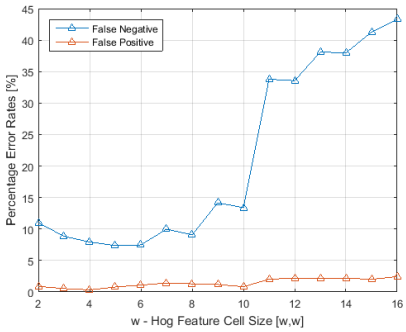
1. HOG
   The default feature extraction methods that we used in our experiments with varying pre- and post- processing parameters was Histogram of Oriented Gradients (HOG). Due to the low resolution of each cell, HOG cell sizes of [8,8] or greater tends to lose shape information of the cells, as is evident when visualizing the extracted features in Figure 3a. The cross-validation result further supports this observation by showing that as cell size gets greater than [8,8], the false negative rate increases significantly.

2. Dense-SIFT
   SIFT is another descriptor that is commonly used for object detection. However, because both the resolution of each cell and the contrast of some cells to the background are very low, the standard SIFT descriptor [3] sometimes refuses to return any significant feature points for a cell window, as observed by visualizing the locations of SIFT features in the training dataset. Therefore, the SIFT features is not suitable as input to the SVM trainer.

   Dense-SIFT, on the other hand, provides a descriptor very similar to SIFT but at every location where a descriptor was calculated, instead of choosing only a few key points [2]. The sizes of extracted features from all windows are thus consistent and can be used directly for training a SVM model.

   The parameter we varied in using Dense-SIFT is the size of a SIFT spatial bin, binSize. In general, both rFN and rFP decrease with decreasing binSize. Decreasing binSize, however, also increases the computational complexity by increasing the total number of length-128 descriptors for each window.

(a)



(b)

Figure 3: 100-fold cross-validation results of HOG feature study: (a) visualization of the extracted HOG features using cell size [8,8]; (b) the error rates by varying cell sizes in one experiment. This experiment was using cropping window size of [32,32] and gaussian filter with size [9,9] and sigma of 1.5.

3. Gabor Filter
   We also experimented with different sizes of gabor filter banks. Gabor filters are often used for edge detection and texture analysis and stereo disparity estimation. We hypothesized that we could extract useful features by filtering inputs with a collection of gabor filters with different scales and orientations. In particular, we used 5 different scales and 8 orientations of gabor filters. In order to select the proper filter size, we evaluated filter sizes from 3 by 3 to 31 by 31 by cross-valiation. We constructed features by concatenating the gabor features of the input image, and the feature vectors are normalized to zero mean and unit variance. Based on the cross-validation, we fixed the filter size to be 16, and features to be extracted from the images of window size 16 by 16. These parameters achieved 1.16% false positive rate and 6.67% false negative rate on the cross-validation.

4. Pre-trained CNNs
   The limited size of our manually labeled dataset (882 cell windows, 6251 no-cell windows) is not

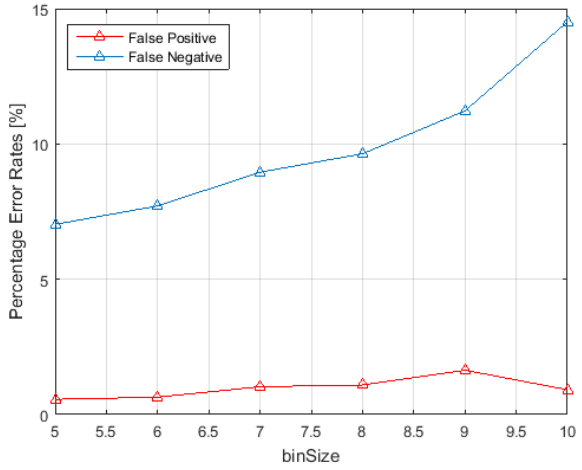


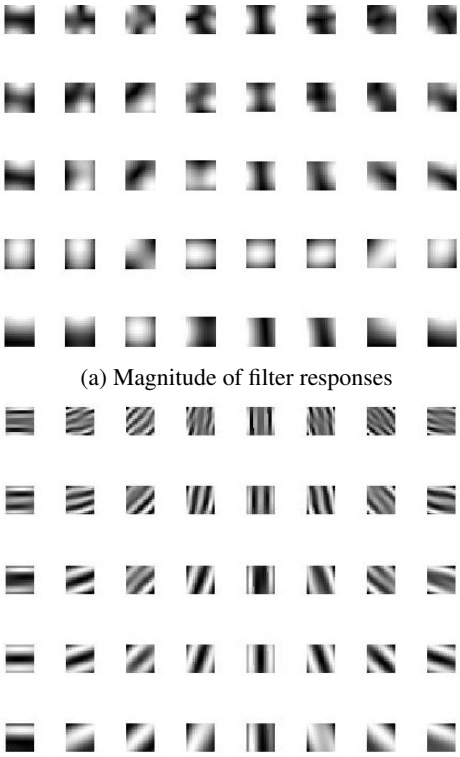
Figure 4: 100-fold cross-validation results of dense-SIFT feature study: rFN decreases significantly with decreasing size of the spatial bin. Among the binSizes that we experimented with, [5,5] had the lowest rFN and rFP rates. This experiment was using cropping window size of [32,32] and gaussian filter with size [9,9] and sigma of 1.5.

enough for training a deep convolutional neural network (CNN). Therefore, we experimented with two pre-trained CNNs, AlexNet [1] and VGG [9].

(a) AlexNet
Ideally, the outputs from the first few layers would be preferable to use; however, the output feature length from the first few layers of the package were on the order of 1e5 to 1e6, which made the computation too complex for our computers to handle. Nevertheless we were able to implement our algorithm using the outputs from layers 16 to 22 of the network, and got some level of success as shown in Table 1. In the future, we can try use a different AlexNet package or a different platform more suited for CNN implementations.

(b) VGG
A pre-trained VGG neural network was used to extract the image features. The last layer of VGG output was inputted into the SVM classifier. The optimized rFP and rFN were 0.78% and 8.4%, respectively. This results was almost comparable to HOG and other feature extraction techniques that we used, considering that no parameters in the VGG were fine-tuned.

5. Summary
Most of the feature extraction techniques we experimented with were able to achieve similar performances in cross-validation after parameter tuning. The performances could potentially be further improved with more extensive tuning of the parameters of the feature extraction scheme, pre-processing steps, and the SVM trainer.



(a) Magnitude of filter responses

(b) Real parts of filter responses

Figure 5: Features extracted by gabor filters at 5 scale and 8 orientations with filter size 16 by 16; column indicates different orientations, and row indicates different scale

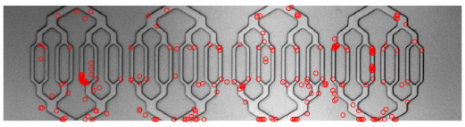
Most of the feature extraction techniques we experimented with were able to achieve similar performances in cross-validation after parameter tuning. The performances could potentially be further improved with more extensive tuning of the parameters of the feature extraction scheme, pre-processing steps, and the SVM trainer.

4.3. Post-Processing

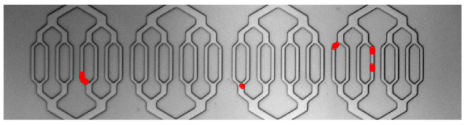
With the location information of the windows labeled as positive by SVM, we removed any positive that is by itself or with few other positives in its adajancy. This process was tuned using the first few frames of a video, based on the fact that the mass majority of such points are false positives. However, we note that this also resulted in some additional false negatives in some frames. Then, we keep only the positive point that has the largest number of adjacent cells

Feature Extraction	False Positive	False Negative	Comments
HOG	0.68%	6.35%	Window size [32,32]; Gaussian [15,15], =2.5; HOG window [5,5]; 100-fold CV
Dense-SIFT	0.56%	0.56%	Window size [32,32]; Gaussian [9,9], =1.5; binSize = 5; 100-fold CV
Gabor Filter	1.16%	6.67%	Subset of no cell training set Window size [16,16]; 5 scales, 8 orientations, filter size 16; 10-fold CV
AlexNet	1.31%	17.57%	Window size [32,32]; Layer 16: pool5'; 10-fold CV
VGG	0.78%	8.40%	Window size [32,32]; Gaussian [15,15], =1.5; averaged 10-fold CV

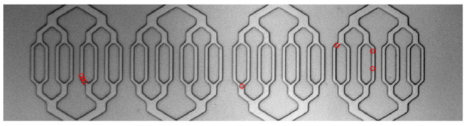
Table 1: Summary of the feature extraction methods' performances



(a) SVM output



(b) Thresholding



(c) Non-maxima suppression

Figure 6: Post-processing: (a) direct SVM output include too many FPs; (b) result after removing any positive that is by itself, has only 1 other positive in a [3,3] window centered on it, or has less than 2 other positives in a [5,5] window centered on it; (c) result after only keeping the positive point that has the largest number of adjacent cells in each cluster.

in each cluster of positives surrounding each cell to obtain [x,y] coordinates of the cell.

4.4. Comparison to Non-ML Image Segmentation Method

After extracting the x- and y- coordinates from post-processing, the results are visualized with manual inputs and non-machine learning image segmentation method. Please see videos 0001 and 0002 in the supplementary. Note that green rectangles represent manual inputs. Blue and red circles represent non-machine learning results and our machine-learning results, respectively. We have observed that there are some true detections that are detected by both algorithm. Our machine-learning method has shown improvement in performance for cells that are clogged and cells that are small and harder to detect by non-machine learning method. However, there are some false positives and missed detections that only appear in our machine-learning method, particularly near the image border. The x- and y- coordinates are also inputted into our tracking algorithm which has been fine-tuned for non-machine-learning method (see supplementary videos 0003 for non-machine learning results and video 0004 for machine-learning results).

5. Conclusion and Future Work

In this report, we have shown that machine learning is a feasible approach towards cell detection. Various feature extraction methods gave low FN and FP rates in SVM cross-validation. The application of using the HOG method on the test video showed promising cell detection rate. However, the cross-frame tracking algorithm need to be improved together with the post-processing steps.

Besides revising the post-processing steps and tracking algorithm, future directions also include increasing the size of the manually labeled dataset to achieve better generalization for SVM training. Non-CNN feature extraction algorithms that detect blobs instead of edges can also be applied since they can potentially give better performance by matching the shape of the cells. We can also try training a shallow neural network using our own dataset.

References

[1] 1994-2016 The MathWorks, Inc. Image category classification using deep learning.

[2] 2007-13 The authors of VLFeat. Dense SIFT as a faster SIFT.

[3] 2007-13 The authors of VLFeat. SIFT detector and descriptor.

[4] O. Al-Kofahi et al. Automated cell lineage construction. *Cell Cycle*, 5(3):327–335, 2006.

[5] O. Debeir et al. Tracking of migrating cells under phase contrast video microscopy with combined mean-shift processes. *IEEE Trans. on Medical Imaging*, 24(6):697–711, 2005.

[6] O. Dzyubachyk et al. Advanced level-set-based cell tracking in time-lapse fluorescence microscopy. *IEEE Trans. on Medical Imaging*, 29(3):852–867, 2010.

[7] Kanade et al. Cell image analysis: Algorithms, system and applications. 2011.

[8] M. J. Rosenbluth et al. Analyzing cell mechanics in hematologic diseases with microfluidic biophysical flow cytometry. *Lab on a Chip*, 8(7):1062–1070, 2008.

[9] A. Vedaldi and A. Zisserman. VGG Convolutional Neural Networks Practical.