Recognizing Hand Gestures with a 3D Camera

Michael Marx, Michael Fenton, Gage Hills

Distinct hand gestures were identified and classified from 3D camera data using multiple machine learning approaches. Two divergent development paths were attempted successfully. The first used simple image segmentation and normalization, followed by feature selection for a support vector machine working off of pixel-based features. The second approach focused on generating more sophisticated features and used an ensemble of bagged decision trees for classification. Both approaches were successful in obtaining a median classification rate higher than 99% for each hand state in the data set.

1. Introduction

3D cameras have the potential to allow new types of user interaction with computers and other devices. Most 3D camera tracking currently used for user input, such as with the Microsoft Kinect system, recognize position of the body at a distance. To our knowledge, no current systems operate in a modality in which sufficient resolution and accuracy are obtained on hands to recognize specific finger gestures from 3D data.

For this project, we used a prototype 3d camera system intended for mobile devices to capture data in a modality in which the hand is captured with high resolution and high fidelity. The objective of the project was then to generate a method of classifying the hand state on a frame by frame basis. Coupled with temporal tracking, this would allow for a robust user interface similar to the "multi-touch track pad" system of using single finger, multiple fingers, and other hand positions to generate unique control inputs.

2. Data Collection

A small 3D camera was used to collect many frames of a user moving their hand with four different hand configurations: one finger point, two finger point, closed fist, and open hand. For each of these gestures, we captured 250 frames as the user moved their hand throughout the field of view while maintaining the gesture. For the context of this document, the data set for a single user's right hand was used.[1] This data set was meant to exercise single hand gesture classification with arbitrary position and orientation.

An individual frame of data is a 320x200 pixel field of brightness and phase values. Brightness corresponds to the amount of light being reflected from the target, and phase corresponds to the distance from the camera to the target. Sample frames of magnitude and phase data are shown below in Figure 1.

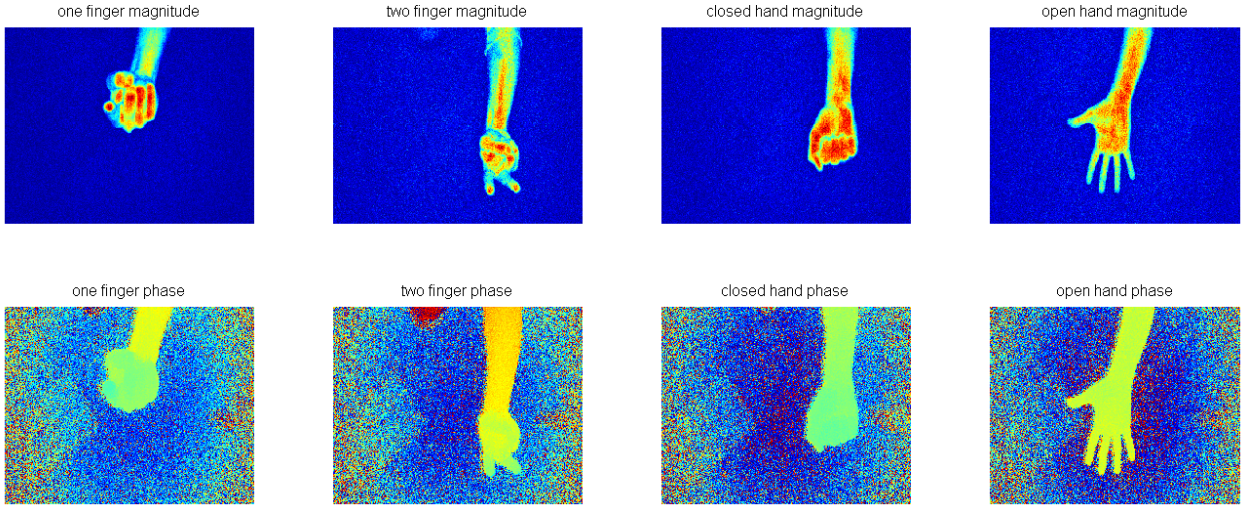
3. Image Processing

Several steps were taken in preparing the data for both classification approaches. The goal of data preparation was to reduce each frame to a consistent representation of the hand without a background. The first step was to segment the full hand and arm in each frame. This was accomplished by removing sufficiently dim pixels, filling any resulting holes in the foreground image (i.e. imclose), then finally labeling the largest contiguous blob as the hand (as it was the largest, brightest object in all of our datasets). Simple inspection was sufficient to show the effectiveness of this mechanism to separate the hand from the background.

In the base data set, the hand occasionally moved outside of the field of view. In order to only classify frames in which the hand was fully visible, we looked for intersections of the hand/arm outline with the frame border and ignored frames which had more than two points of overlap.

[1] In order to explore the ability to classify across left and right hands and across different users, other data sets were collected using different users and alternating hands, however these results have been omitted for brevity.

Figure 1: Magnitude and Phase of 4 Hand Positions



Additionally, we identified which row contained the wrist (using a simple function of foreground thickness variation across rows) and removed frames that did not have a wrist as defined by this constraint. After these removal stages, 896 frames of full hand data were preserved with pixel-wise determined positions of both the hand and wrist.

4. SVM-based Classification

The first of the two major approaches to classification taken in this project was to apply Support Vector Machines to learning how to classify a particular hand state. This was initially attempted using the raw magnitude and phase data, as well as a few manipulations and features extracted from the data. However, it was found that the large variability in hand location and orientation defeated the SVM's attempts at classification. Focus was instead given to separating the hand from the rest of the image, and to generate a normalized and reduced set of features for the SVM to regress against.

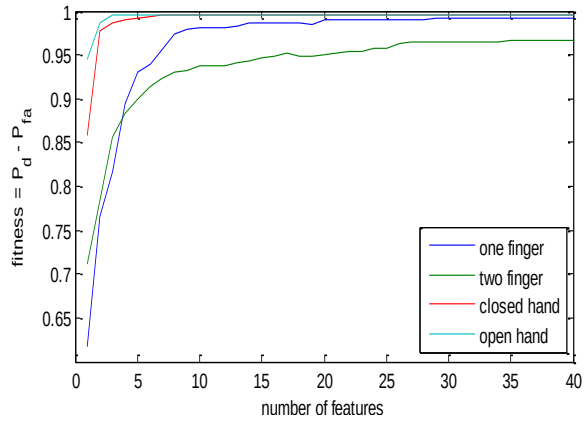
With the hand locations determined for every frame, the data was normalized for SVM application. It was desired that distance from the camera (and related scaling of hand size) would not throw off the classification. To enable this, each frame was resampled down to a consistent field (100x100 pixels) that spanned from the wrist to the farthest fingertip and from the far left to the far right side of the hand. Further, we separated the hand from the background by renormalizing the distances based on the minimum object distance.

Using the normalized distance per pixel as the input to SVM, the classifiers were able to achieve an overall accuracy of more than 98%, (using 90%/10% train/test cross-validation,) but took an exceptionally long time to compute. In order to reduce computation time, feature selection was employed.

To determine performance against a small number of features, four sets of 75%/25% train/test cross-validation were prepared. All features were tested for each of the four classifiers, with a fitness function defined as (fitness) = (probability of detect) – (probability of false alarm) for each. (These will be expressed as P_d and P_{fa} henceforth.) At each stage, each unused feature was fed to an SVM along with the previously determined best features, and the new feature that achieved the best fitness value was then added to the list of best features. The available features were the previously described normalized phases along with many other features that ended up being discarded by the feature selection process. Performance of the SVM for each of the 4 cases against different numbers of features is shown in figure 2. Maximum performance (among the number of features computed) was achieved with a minimum of 29 features for one finger, 35 features for two finger, 7 features for closed hand, and 3 features for open hand.

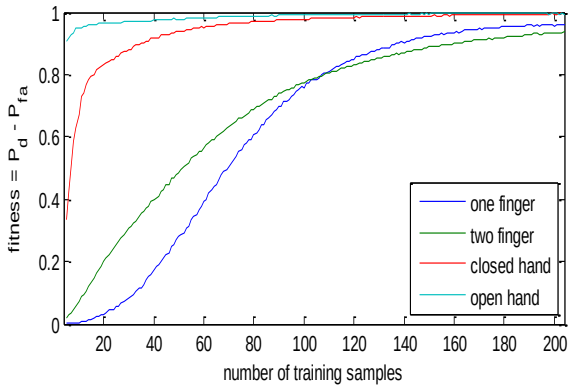
In order to determine an upper limit on the effectiveness of the selected features, leave one out cross validation was employed using only the selected features for each classifier. One finger classification achieved 99.55%/1.18% P_d/P_{fa}. Two finger classification achieved 96.37%/0.62%. Closed hand and open hand were classified with 99.02% and 99.55% detection rate respectively, and both had zero false alarms. Across the full data set, every frame was either classified correctly, not classified, or classified as two possible hands. In no case (during LOOCV) was a frame classified incorrectly.

Figure 2: Fitness Function vs. Number of Selected Features



Finally, the amount of training necessary was investigated. Different portions of the data was selected for training the SVM and used to test against the remaining data. The resulting curves are shown in figure 3. As can be seen from the curves, the open hand case is very easy to classify, achieving 95% fitness after only 10 samples, and reaching 99% fitness after 90 samples. The closed hand classifier also converges quickly, reaching 80% fitness in only 17 samples, and reaching 99% accuracy in 90 samples. The one and two finger cases take much longer to converge, both taking 109 samples to reach 80% fitness and growing slowly from there. It is believed that this result reflects the SVM requiring training frames that are similar to the test frames in order to classify accurately. As the training set size grows, it becomes increasingly likely that neighboring frames to the test data will have been included in the training data. Because the hand moves continuously in space and time, the nearby frames will be more similar than disparate frames.

Figure 3: Fitness Growth vs. Size of Training Set

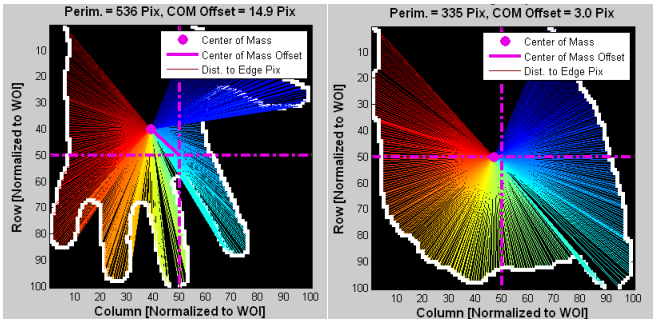


5. Ensemble-based Classification

For the ensemble method our approach was to build up as many features as possible that provided some intuitive information about the hand's position. These features need to be only slightly more descriptive than 50% in being able to distinguish between any two gestures to be helpful. Additionally, the more distinguishing the feature is, the more useful the feature is to the classification process.

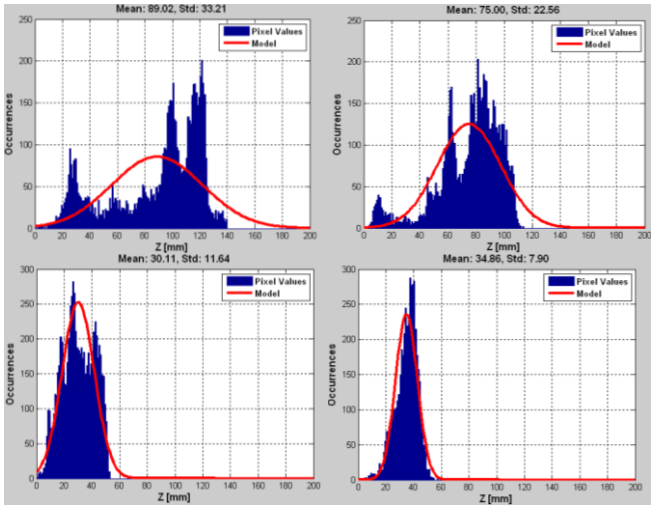
The advantage of tree-based classification is that it does not require a single feature to clearly discriminate one gesture from another; with sufficient features, it is able to optimize the decision tree to maximize the classification rate, even in the presence of fairly weak features. Better features will likely also lead to a more compact decision tree. For example, a highly descriptive feature is the perimeter of the hand (in either pixel distance, number of edge pixels, or in XY space) which separates the "open hand" gesture well from the other three gestures. Consequently, the first decision in the tree is based on the perimeter of the hand; a perimeter above a certain threshold automatically classifies into the "open hand" gesture. The effect of the perimeter is seen in the comparison below.

Figure 4: Edge Map of Open and Close Hand



Another useful feature is the distribution of depth values of each of the four gestures; the variance is a strong indicator of which label the gesture belongs to, pictured below.

Figure 5: Depth Histogram & Variance Plot (Two Finger Point, One Finger Point, Closed Fist, Open Hand)



Other key features that we used are described in the table below, as well as qualitative description of their expected behavior with respect to each gesture.

Feature	***Expected behavior***
Perimeter/variance in radius from center of mass	Distinguish open hand from other three gestures (extra edge pixels along fingers instead of just around hand)
Center of mass	Closed fist to have center of mass closest to center of WOI, open hand to have center of mass farthest away
Variance/range in Z	Open hand to have little variance in Z (hand is along a single plane in Cartesian space), two-finger point to have highest variance (two fingers with closer depth than hand)
Number of pixels closer than Z_{THRESH} closer than the max distance on the hand	Distinguish one finger & two finger point, as fingers should be greater than roughly 50mm from the center of the hand
Horizontal gradient in Z	Distinguish one finger & two finger point, as there are more local depth changes than on flat open hand

In order to evaluate the expected performance of each of our features, we used a metric to measure the normalized distance between two distributions. For each feature, that feature's value is calculated for each frame (e.g. the depth variance for the hand, in the case above) and lumped into one of four distributions based on its label. The figure of merit for that feature measures the *maximum* distance between all six combinations of the four distributions. The maximum is representative of the feature strength because a bag tree does well if it can separate any two of the features. We used the following metric to determine the expected performance of feature *f*, for gestures *i* and *j*:

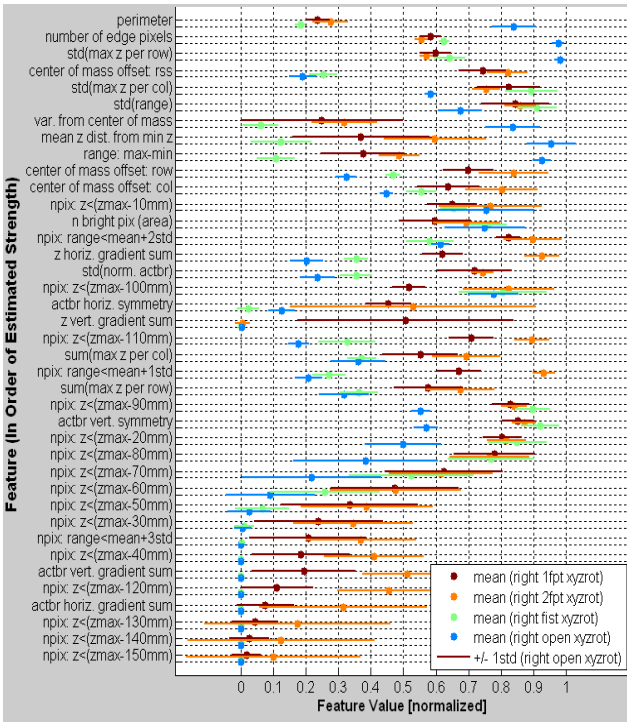
$$\delta_f = max_{i,j}\left(\frac{|\mu_{fi} - \mu_{fj}|}{\sqrt{\sigma_{fi}^2 + \sigma_{fj}^2}}\right)$$

Sorting the features by this metric yields the table in Figure 6.

We chose the features as they have an intuitive correlation to the gestures. However, we found that we were also able to use features that have a less intuitive relationship with the gestures when observed independently. Take, for example, the absolute depth measurement of the hand. There is not a reason why this feature alone should be able to classify the gesture, since any gesture could be performed at any depth. The absolute depth measurement can, however, identify proper thresholds for classification with other features such as the perimeter or area of the hand – a closed fist may have the

same perimeter as a far away open hand, thus the absolute depth helps to separate cases such as these in conjunction with other classifiers.

Figure 6: Feature Descriptions & Distribution



For these reasons, we found the best results when we fed as many features as possible to the classification tree builder, and let it optimize based on what it found most useful, as opposed to only feeding it features we thought would be useful.

Using these features, we used the ensemble method of bagged decision trees to classify each of the frames of data, randomly subsampling the dataset into 75%/25% training/test subgroups. The resulting classification rate for 100 trees per subgroup was over 95% in all cases, with a median classification rate of 99.33% across the four gestures. This surprisingly good success rate prompted us to expand the dataset to include five users and both hands for those users; the median classification rate was 95% across the four gestures over all of the expanded dataset, which shows both reasonable success and promise for future improvements.

6. Conclusions

We were successful in creating two classifiers which were able to accurately recognize hand configurations using 3D camera data. Both approaches relied heavily upon properly segmenting the image, which is a non-trivial task for traditional RGB cameras, but was relatively simple for the case of our 3D camera. We were able to attain over 99% successful classification rate for both the SVM-based and Ensemble Method-based classifiers on our dataset.

The per-pixel SVM-based approach would be an ideal implementation for high performance applications which required minimal classification times and did not require updates to the support vectors frequently (due to the extraordinarily slow process of identifying support vectors in our datasets).

The Ensemble Method-based classifier had significantly higher per-frame runtime requirements (due to the computation involved in calculating the feature values), but a much lower regeneration time, which would make it ideal for situations where conditions or users were frequently changing.

Using either of our hand classification systems, it would be straightforward to implement gesture recognition for a user interface. By giving an immediate classification of hand state in every frame, a second level program could monitor the hand state and position continuously to recognize a set of control inputs, enabling a more robust level of interaction for the user.