Hand Gesture Recognition in Images and Video

Ilan Steinberg, Tomer M. London, Dotan Di Castro
Department of Electrical Engineering, Technion- Israel Institute of Technology

Abstract- As the integration of digital cameras within personal computing devices becomes a major trend, a real opportunity exists to develop more natural Human-Computer Interfaces that rely on user gestures. In this work, we present a system that acquires and classifies users' hand gestures from images and videos. Using inputs from low resolution off-the-shelf web cameras, our algorithm identifies the location and shape of the depicted hand gesture and classifies it into one of several predefined gestures. Our algorithm first applies image processing techniques on the images in order to cancel background and noise effects on the image, it then extracts relevant features for classification and finally classifies the gesture features using a multiclass Support Vector Machine classifier. The algorithm is robust and operates well on several different backgrounds, lighting and noise conditions. Our method achieves an average 97.8% accuracy rate in several cases and it is suitable for both real-time and offline classification.

1. INTRODUCTION

Digital cameras are now integrated into personal computers, mobile cellular devices and handheld computers. These devices usually include a powerful microprocessor, capable of performing millions of computations per second. As microprocessor and digital camera technologies advance, it is now possible to use these resources in order to create new human computer interfaces that are based on recognition of users' gestures. Gesture recognition interfaces can be used as a natural communication channel between humans and machines and give rise to a plethora of applications such as hardware-free remote controls, sign language interpretation [8] and other human welfare applications [1].

In order to be applicable to current PCs and mobile devices, a gesture recognition system should be based on existing and common hardware such as low-resolution web cams or mobile-integrated cameras. It is also desired that the system will be able to operate under non-uniform background, lighting and noise conditions. Another requirement for the gesture recognition system is to be computationally non-intensive in order to be suitable for real-time classification.

In this paper, we present a system that acquires and classifies users' hand gestures from images and videos using image processing techniques and machine learning algorithms. This system includes a low-resolution webcam and an algorithm that processes the acquired images and classifies the gestures presented. The system contains a setup procedure that executes once, where the classifier is trained on a given training set that contains examples of the different gestures. After the setup is done, the system is ready to classify input images and videos.

Given an acquired image that contains a hand gesture, the hand gesture recognition algorithm performs two stages. The first is the preprocessing phase, where the hand shape and other distinguishable features are extracted from the image using noise reduction, filtering and several other image processing techniques. The second stage involves the classification of the features set to the appropriate gesture using the trained classifier that was computed in the setup stage. In order to classify an image given many different possible gestures, we use multiclass classification that relies on several binary classifiers. Specifically, we use Support Vector Machines (SVM) [9] as binary classifiers and integrate their binary classifications to a single multiclass classification using a "Classifier Tournament" approach. We show that this algorithm is robust and operates well

---

• *Ilan Steinberg: ilanst1@gmail.com*
• *Tomer M. London: tomerlondon@gmail.com*
• *Dotan Di Castro: dot@tx.technion.ac.il*

on several different backgrounds, lighting and moderate noise conditions and it is suitable for both real-time and offline classification.

The rest of this paper is organized as follows: In section 2, we discuss previous work on hand gesture recognition. Section 3 describes the system architecture. In section 4 we present the preprocessing algorithm while in section 5 the classification method is presented. In section 6 we show the classification results and statistics of our system. Finally, in section 7 we discuss future work and in section 8 our conclusions are presented.

2. Related Work

Several different approaches have been used in order to design hand gesture recognition systems. Notable approaches involve using input from special electronic gloves [2] or using input from specially marker gloves, or marked hands [3][4][5]. The inconvenience of marker-based systems makes them unattractive for every-day use as human computer interfaces. Other systems are based on the extrapolation of complex representation of hand shapes [6]. This approach involves complex computations and therefore is unattractive for real-time and computational bounded applications.

A marker-free, visual hand recognition system was proposed in [10][7], where the classification is performed in the curvature space. This approach involves finding the boundary contours of the hand and it is robust in scale, translation and rotation, yet it is extremely demanding computationally. In [11] a multi-system camera is used to pick the center of gravity of the hand and points with maximal distances from the center provide the locations of the finger tips, which are then used to obtain a skeleton image, and finally for gesture recognition. In [23] a special camera that supplies dept information was used to identify hand gestures. Other computer vision methods used for hand gesture recognition include specialized mappings architecture [13], principal component analysis [14], Fourier descriptors, neural networks, orientation histograms [15], and particle filters [16].

Unlike previous work, we focus on hand gesture classification using inputs from low-resolution digital cameras and classify the acquired images using features extracted by non-computationally intensive image processing techniques. We classify the extracted feature sets using a multiclass SVM that was trained in a setup phase and is specifically trained to distinguish between predefined gestures. The system operates in moderately noisy environments and in non-uniform backgrounds and is suitable for both real-time and offline classification.

3. ARCHITECTURE

The algorithm presented in this paper has two main stages, a preprocessing stage and a classification stage. The preprocessing stage prepares the input image from the camera and extracts features for the classification stage. In the classification stage a support vector machine (SVM) (see 5.2) algorithm is used to classify the features extracted to a known hand gesture. The SVM algorithm requires a setup procedure, before it can be used for classification, where the multiclass SVM algorithm is trained. The training is done using examples of the different gestures shown in either videos or images. In the case of a video, it is broken down to frames and each frame converted to an image. The system runs the images through the preprocessing stage and extracts the relevant features from each image. The features and the classification of each input image are used to train the multiclass SVM algorithm to correctly classify the gesture in that image. The setup procedure runs only once, after the system is trained it is ready to classify.

The classifying procedure can be a classification of an offline video showing hand gestures or classification of a real-time online camera video showing hand gestures. In both cases the video is broken down to frames and the frames converted to images. The images are then preprocessed and the relevant features are extracted. These features are passed onto the multiclass SVM and a hand gestures classification is made.

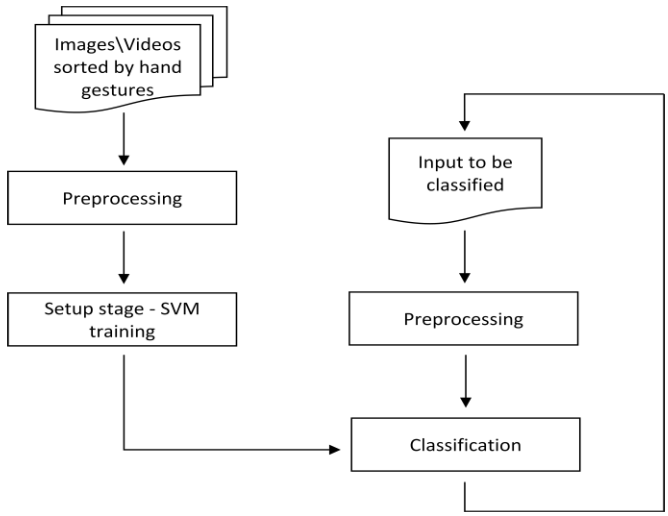

Figure 1 - System architecture

4. PREPROCESSING

As mentioned in the introduction the main purpose of the preprocessing stage is to extract the hand and features of the hand from the input image and pass them on to the pattern recognition stage, which we discuss in the next section. The preprocessing stage is of great importance to the success of the classification, if we fail to extract a proper hand figure and relevant features of the hand from the input image, the pattern recognition algorithm won't be able to achieve the desired results. Furthermore, it is vital that the preprocessing part will be as robust as possible so environment changes such as background, noise and lighting won't affect the classification result.

We will now explain the methods we took to insure that the preprocessing stage will produce robust and accurate results. A diagram of the system is shown in *fig*. 2:

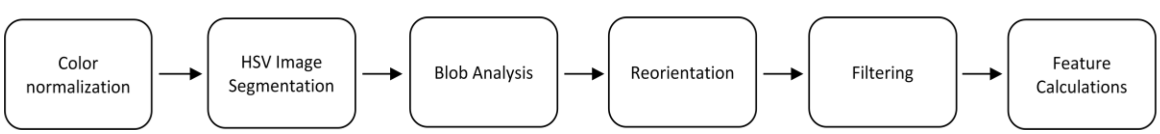

Figure 2 - The preprocessing algorithm stages

4.1 Color Normalization

As we show in the next several stages, the preprocessing algorithm we use is highly dependent on color analysis. We have stressed the importance of robust preprocessing, thus we show how we maintain robustness in different light conditions.

Different light conditions may impair our robustness because of the algorithm's dependencies. By simply subtracting the mean of each color in the RGB picture we cancel the effect of lighting color and compute approximately the same image for different lighting colors as may be seen in *fig*. 3:

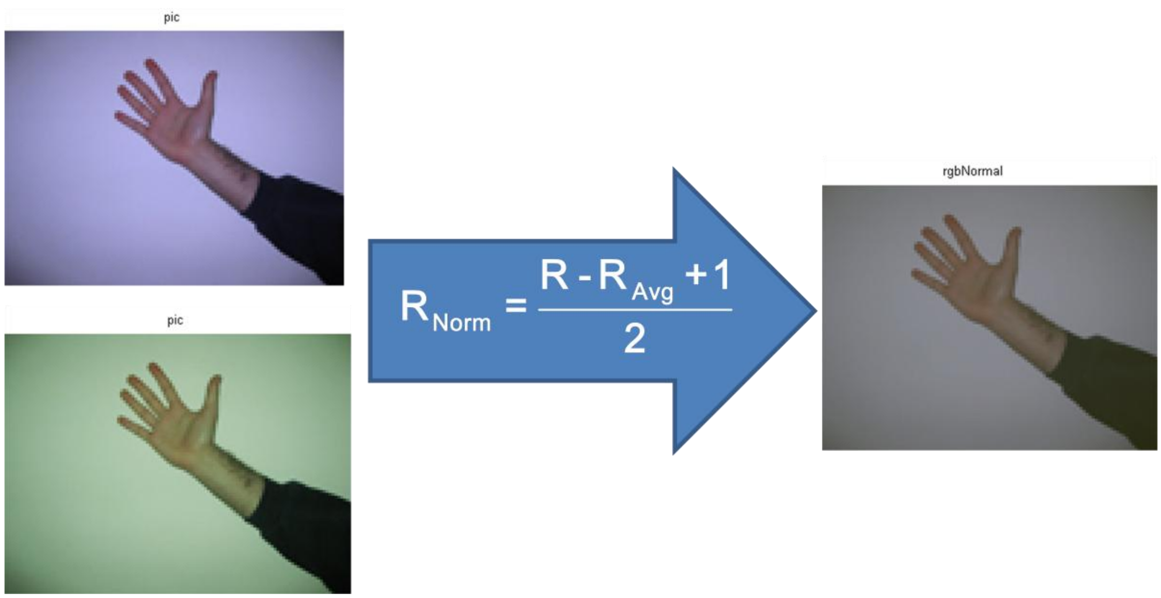

Figure 3 - Color normalization

This normalization makes sure that the normalized values are in the proper range of the RGB image. This method has proved effective and helped maintain the high robustness of the system, however, this is under the assumption of a background that does not hold color components similar to the components of the hand color. If this was done on a red background picture we might not get the same results.

4.2 Image Segmentation Based On Color

By default, an image is represented in an RGB format, meaning that for each pixel in the image each of the colors Red, Green and Blue has a value stating the intensity of that color in the pixel. Since different people have different hand colors (e.g. Black hands vs. white hands) we cannot rely on the intensity of any one of the colors in order to extract the hand and its features. Therefore we converted the RGB representation to a Hue-Saturation-Value (HSV) representation, meaning that now for each pixel in the image there are three values – Hue, Saturation and Value. By looking on different hand pictures and measuring their HSV values we concluded that we can find hue and saturation thresholds that have high correlation with typical hand colors. We found that typical skin HSV values are: $0.12 \le S \le 0.4$, $0.93 \le H \le 1$ or $0 \le H \le 0.137$.

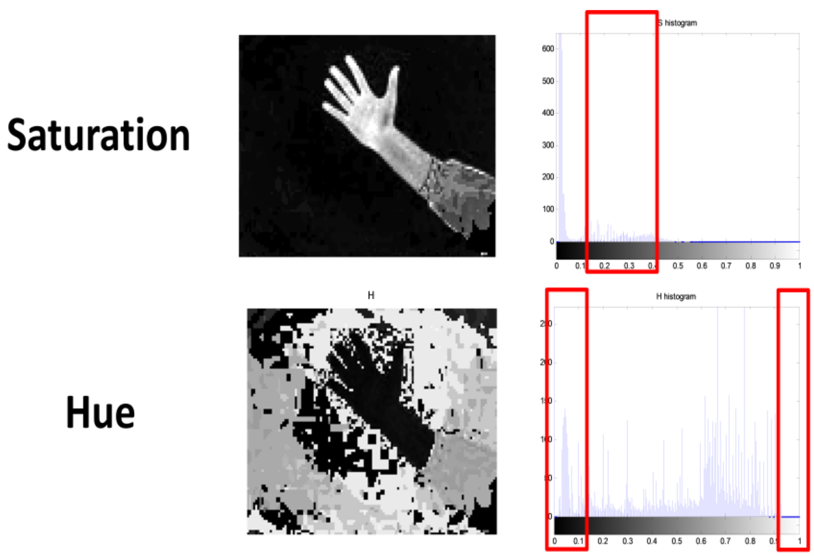

Figure 4 – HSV image segmentation

In *fig.* 4, we show an example of HSV values of a hand image. The values marked in red are the skin HSV values and pixels these values are selected to move to the next stage. In *fig.* 5, the pixels with the skin HSV values are colored in white while the rest of the picture is colored black.

4.3 Blob Analysis

After finding the areas suspected of being a hand, we now must choose between these areas. We find all the connective parts of the picture and choose the biggest (largest area) amongst them (since we assume the hand is the largest area suspected of being a hand). All other areas are colored in black and the BW picture (*fig.* 5) is a black and white binary image of the hand. If you cannot make the above assumption you may want to use an SVM learning machine (which we will discuss in detail in the pattern recognition section) to check whether each suspected area is or isn't a hand, this solution can also satisfy the problem that may rise if you have more than one hand in the picture. Now that we have only one hand area we continue to process this area in the picture.

4.4 Orientation

In different pictures the hand may have very different orientation, this may damage the image recognition results and therefore should be dealt with in the preprocessing stage. We deal with this problem by finding the main axis of the hand (the longest axis), calculating its orientation and reorienting it to have an orientation of zero degrees. The results of such reorientation are that similar hand gestures in different orientation are almost exactly the same and are much easier to classify. *Fig.* 5 demonstrates the blob analysis and the Orientation processes we discussed in the previous sections:

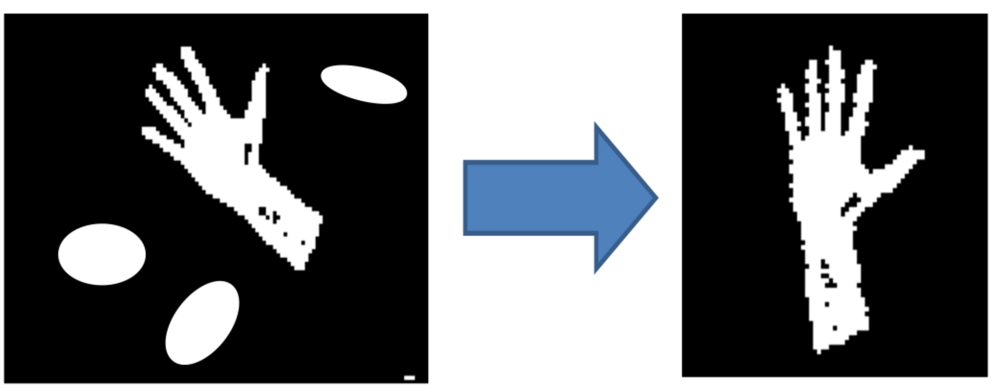

Figure 5 - BW picture

4.5 Filtering

We explained the extraction of the BW picture from the original picture. Even though the above process extracts the hand successfully, it may still create holes or rough edges. We use a median filter in order to fill these holes and sharpen the edges. A median filter is a filter that assigns each pixel with the median value of its neighbors. In a black and white picture it gives the pixel the same color as the majority of that pixel's neighbors. The results of this filtering can be seen in *fig.* 6:

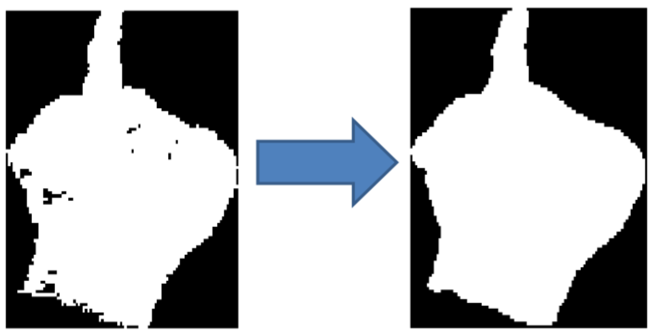

Figure 6 - Median filter

4.6 Feature Calculations

In this part we extract features of the hand which will be passed on to the gesture recognition stage to classify. Having more features may improve the gesture recognition quality but having too many features may induce over-fitting and reduced robustness. Having too many features may also incur complexity problems and worsen the SVM algorithm convergence time. Therefore, we reduced the dimensions of each picture to 100x100 pixels and used each pixel as a feature to pass on to the next stage.

By filtering, normalizing, manipulating and defining thresholds values we have shown how we preprocessed an input image and extracted its features for classification while maintaining a robust and accurate output. In the next section we use these features to indentify the gesture the hand is showing using an SVM algorithm.

5. GESTURE RECOGNITION

In the previous section we have shown how we extract the features from the image, now we use these features to classify the gesture the hand is showing by using a multiclass SVM algorithm. We first provide a brief introduction to Machine Learning and then present the principals of SVM classification.

5.1 Machine Learning

Machine Learning algorithms have the ability to learn from experience — that is, to modify their execution on the basis of newly acquired information. A typical machine learning algorithm uses given examples to develop classification capabilities or decision making capabilities in order to operate on new unknown inputs. There are three types of learning algorithms:

- *Unsupervised Learning* – the machine is only given a set of examples (training set) but no classification for these examples is given. This algorithm type is used for clustering problems.
- *Supervised Learning* – the machine is given a set of examples and the correct classification for them (this is called the training set). This algorithm type is used for classification and regression problems.
- *Reinforcement Learning* – no examples are given but positive or negative reinforcement is given after each decision is made. This algorithm type is used mostly for control problems were there is feedback such as winning or losing a game.

5.2 The Support Vector Machine Algorithm

The SVM algorithm is one of many algorithms for supervised learning. Generally, an SVM is a linear classification algorithm that maximizes the distance between the decision line (discriminator) and the closest example to it in the training set. An illustration is given in *fig. 7*. Generally an SVM is used to classify only two groups.

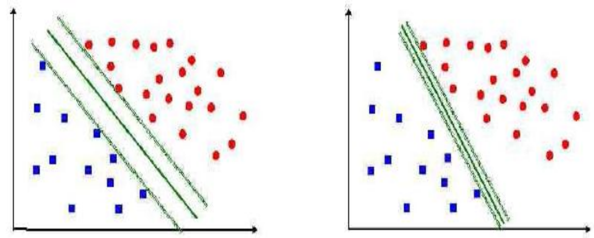

Figure 7 - Linear discriminators (taken from [22])

5.3 Theory of SVM

We label the training set $\{x_i, y_i\},\ i=1,..,n,\ x_i \in R^d,\ y_i \in \{-1,1\}$, where training example i's features are represented in the $d \times 1$ feature vector x_i and the example's correct classification is y_i. We call a group of training points 'linearly separable' if there exists a linear hyperplane that divides R^d in a way that all the points of classification '1' are found on one side of the hyperplane while all the points of classification '-1' are found on the other side of the hyperplane. A hyperplane is characterized with a *direction vector* w and an *offset vector* b that satisfy the equation $\sum_{i=1}^{d} w_i x^i + b = 0$, where d is the dimension of the plane, or the vectored equation $w' x + b = 0$, where ($w' = w^T$).

The training group $\{x_i, y_i\}_{i=0, 1..., n}$ as described above, is linearly separable if there exists a linear hyperplane $w' x + b = 0$ that satisfies:

$$\text{sign}(w' x_i + b) = y_i \quad i = 1,2, \ldots, n.$$

The distance $d(x_0)$ between a point x_0 and the plane $w' x + b = 0$ is given by

$$d(x_0) = \frac{w' x + b}{\|w\|} \quad (4).$$

The absolute value of this distance is the Euclidean distance of the point from the hyperplane and the sign of $d(x_0)$ is the side of the hyperplane the point rests, where a positive sign means the point rests in the direction of w (w is orthogonal to the plane) from the hyperplane and for a negative sign vice versa.

Now let us remember that the goal of SVM is to **maximize** this distance for the **closest** point to the hyperplane, so we need to solve equation (5),

$$\max_{w,b}\left\{\frac{\min_k\{y_k(w'x_k+b)\}}{\|w\|}\right\} \quad (5).$$

Any set of parameters (w, b) can be normalized by a definite constant without changing the hyperplane. Let us choose to normalize them in such a way that $\min_{1\leq k\leq n} y_k(w'x_k + b) = 1$ then we will have to maximize

$$\max_{w,b}\left\{\frac{1}{\|w\|}\right\} \quad \text{subject to} \quad \min_{1\leq k\leq n} y_k(w'x_k + b) = 1 \quad (6).$$

This is equivalent to optimizing

$$\min_{w,b}\left\{\frac{1}{2}\|w\|^2\right\} \quad \text{subject to} \quad y_k(w'x_k + b) \geq 1, \ \ k = 1,2 \ldots ,0 \quad (7).$$

This is a convex optimization problem with linear conditions. This kind of problem can be solved efficiently by a set of numeric optimization algorithms, e.g. Gradient Descent[17].

We gave a basic overview on SVM, in the next few sections we will show how we use it as a nonlinear discriminator using kernels and show how we expanded it to classify between more than two classes.

5.4 Non-Linear Classification

In order to explain the use of kernels for nonlinear classification we must first consider the *dual problem*. In particular, we may state the following *duality theorem* [19]:

If the primal problem has an optimal solution, the dual problem also has an optimal solution, and the corresponding optimal values are equal.

For reasons we will later show, the dual problem is much easier to work with. In order to go from the primal problem to the dual problem we must use the method of Lagrange multipliers [19] on the primary problem. Then use *Kuhn-Tucker conditions* [19][18]. An elaboration can be found in [20].

The dual problem is given in equation (8)

$$\max_{\alpha} \sum_{k=1}^{n} \alpha_k - \frac{1}{2} \sum_{k,l=1}^{n} \alpha_k \alpha_l y_k y_l \cdot x_k{}' x$$

$$\text{s.t.} \quad : \quad \alpha_k \geq 0, \quad k = 1, 2, \ldots\ldots, n; \ (8)$$

$$\textstyle\sum_{k=1}^{n} \alpha_k y_k = 0.$$

Let x_k denote a vector of dimension d and let: $\{\varphi_j(x)\}_{j=1,\ldots,n}$ denote a set of nonlinear transformations from the input space to the features space where m>>d.

We now want to classify using the function: $\hat{f}(x) = sign\big(w' \varphi(x)\big)$ with the relevant coefficients w. In order to find w, we now need to maximize the distance in the feature plane, given by

$$d(x_0 x) = \frac{w' \varphi(x_0)}{\|w\|} \quad (9).$$

We now repeat the solution shown in 5.3 only with $\varphi(x)$ instead of x and $K(x,z) \equiv \langle \varphi(x) | \varphi(z) \rangle$ instead of $\langle x|z \rangle$. We get the optimization problem (10)

$$\max_{\alpha} \sum_{k=1}^{n} \alpha_k - \frac{1}{2} \sum_{k,l=1}^{n} \alpha_k \alpha_l y_k y_l K(x_k, z_l)$$

$$\text{s.t.} \quad : \quad \alpha_k \geq 0, \quad k = 1, 2, \ldots\ldots, n; \ (10)$$

$$\textstyle\sum_{k=1}^{n} \alpha_k y_k = 0.$$

Note that by using the kernel function $K(x,z)$ we can solve the optimization problem to find the separating hyperplane without explicitly calculating $\varphi(x)$- this saves a lot of computation effort because $\varphi(x)$ may be of a very high dimention. After finding the coefficients α_k we can calculate $w = \sum_{k=1}^{n} \alpha_k y_k \varphi(x_k)$ and therefore get the solution

$$\boxed{y = sign\big(w' \vec{\varphi}(x_k)\big) = sign\big(\textstyle\sum_{k=1}^{n} \alpha_k y_k K(x_k, x)\big)}.$$

We showed how to classify linear and nonlinear data sets into two groups (binary classification), we will now show how we combined binary SVMs to build a multiclass SVM.

5.5 Multiclass SVM

Since we need to classify between several hand gestures (and not only two) we combined several binary SVMs in a *Classifier Tournament* structure. For every pair of classes, we train a binary classifier that classifies between them. After new image features are computed in the preprocessing stage, the features are entered into all the binary classifiers. Each binary classifier returns a class number representing a gesture. The gesture that has the most “votes” of classifiers wins and is the result of the multiclass SVM.

Our experiments showed that the advantage of this combination over other possible combinations of binary SVM, such as all vs. all (where each binary SVM checks if one gesture), is that in this method each binary SVM is much more accurate and therefore, so is the final result. Further comparison between different multiclass SVM can be found at [21].

6. RESULTS

In previous sections, the structure of the system was described, the setup, preprocessing and classification stages were elaborated upon and the methods to achieve their purpose were discussed. Now, we will show results that seal these methods as effective methods for hand gesture classification. The process of multiclass SVM training includes the training of several binary SVMs. As we trained the binary SVMs, we measured each binary SVM’s performance rate by using the method of cross validation. Later, a sample video was made showing different hand gestures. The system classified the sample video and the performance was measured yet again. The system was also tested under real time conditions by connecting the computer to an online camera and testing the performance for the third time.

The system was trained using five training movies showing different hand gesture. The hand gestures differ in the number of fingers raised, one finger raised up to five fingers raised. All binary SVMs showed 100% accuracy in the cross validation.

For the online classification the total Accuracy rate was 72% this can be improved and is discussed in the future work section. The lower accuracy rate of the online classification, compared with the offline classification, is due to the use of a lower resolution camera in the classification process in testing of the online accuracy.

In the sample movie the accuracy rate was measured per hand gesture and in total. The results prove the effectiveness of the combination of methods used. In Table 1 the classification accuracy of each hand gesture can be seen as well as the totals, the gesture number corresponds to the number of fingers raised.

Gesture number	Accuracy rate	Number of frames
1	100%	19
2	89.83%	59
3	100%	67
4	100%	93
5	98.94	94
Total	97.8%	332

Table 1 - Hand gesture classification accuracy

As can be seen in *fig. 8* the accuracy of the classification varies as a function of the number of gestures the system has to classify between.

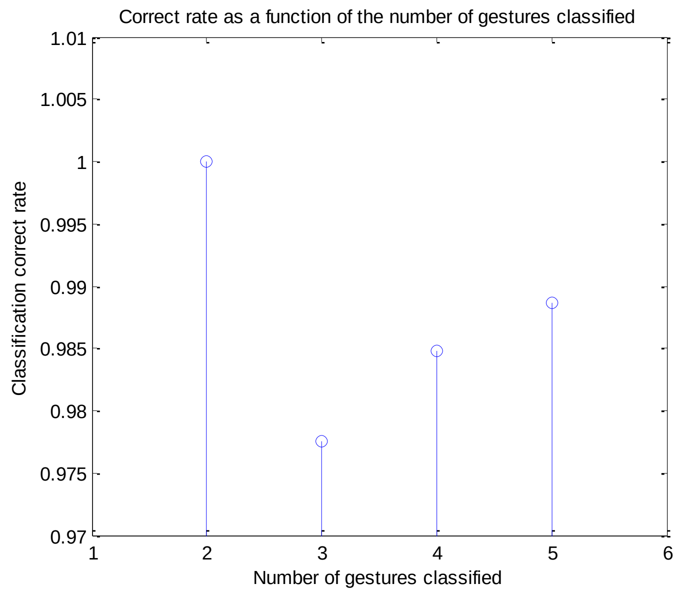

Figure 8 - Correct rate vs. Number of gestures classified

In Table 1 - Hand gesture classification accuracy we see that while classifying between five gestures the most errors occur in the classification of the second gesture, this error is due to the high resemblance between the second and third gestures as can be seen in *fig.9*.

This can also be seen in Figure *8* - Correct rate vs. Number of gestures classified, while there are only two gestures the classification is perfect but when the third gesture is also taken into account the accuracy is the lowest. When adding the fourth and fifth gestures the classification correct rate grows. Hence, the errors that occur are mostly due to the classifier between the second and third gestures.

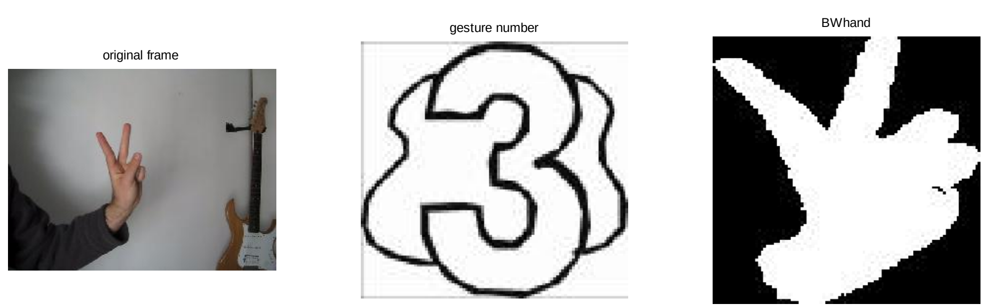

Figure 9 - example of missclassification

7. FUTURE WORK

We have shown a system for the identification and classification of hand gestures from low resolution input images. Future work includes the implementation of our algorithm in system-level programming language in order to increase the efficiency and computational performance of the classification performance. This system can be implemented into PC environment as well as into mobile device and embedded environments.

Further improvement of the classification accuracy may be achieved using an optimized set of features. Several additional features can be extracted from the image and be used in the classification such as hand skeleton extraction, computation of number of raised fingers using a bounding circle around the center of mass of the hand and more. In addition to the spatial information we used in our work, temporal information may be used to complete the feature set. Important data and probability inference can be extracted by looking at the previous frames when classifying a new frame. Accuracy can also be improved by optimizing the binary SVM classification parameters such as training set size, training examples ratio and minimal margin requirements.

Improvement in the computational performance of the algorithm may include using specialized multiclass classifier or hierarchical model classification instead of the "binary classifiers tournament" based approach. Other approaches may prove to be more efficient computationally. Performance can also be boosted by reducing the feature set size by identifying and discarding features that don't contribute a lot of information to the system, algorithms such as Principle component Analysis (PCA) [24] may be used.

Our system can also be integrated and be used for multiple-hands gesture recognition, where each image may include several hands. Our algorithm may be activated several times on each image in order to classify each hand. Another interesting extension to our algorithm is the development of a classifier of multi-stage gestures, which include the recognition of a known sequence of gestures as appears in the American Sign Language.

8. CONCLUSION

The trend of integration of digital cameras within personal computing devices give rise to an important opportunity to develop more natural Human-Computer Interfaces that rely on user gestures. We presented a system that acquires and classifies users' hand gestures from images and videos.

Our algorithm identifies the location and shape of hand gestures and classifies them into one of several predefined gestures. The input of our system is images acquired from low resolution off-the-shelf web cameras. Our algorithm first applies image processing techniques on the images in order to cancel background and noise effects. It then extracts relevant features for classification such as area, orientation and normalized image pixels data and finally classifies the gesture features using a multiclass Support Vector Machine classifier. The algorithm is robust and operates well on several different backgrounds, lighting and moderate noise conditions. Our method achieves an average 97.8% accuracy rate in several cases and it is suitable for both real-time and offline classification.

9. REFERENCES

[1] A. P. Pentland, "Smart rooms: machine understanding of human behavior," in Computer vision for human-machine interaction (R. Cipolla and A. Pentland, eds.), pp. 3–21, Cambridge University Press, 1998.

[2] Dataglove model 2 operating manual. VPL Research Inc., 1989.

[3] V. C. Tartter and K. C. Knowlton. "Perception of sign language from an array of 27 moving spots". Nature, (239):676{678, Feb. 19, 1981.

[4] R. Cipolla, Y. Okamoto, and Y. Kuno. "Robust structure from motion using motion parallax." In Proc. 4th Intl. Conf. Computer Vision.

[5] B. Dorner. "Hand shape identi_cation and tracking for sign language interpretation". In Looking at people workshop, Chambery, France, 1993. IJ-CAI.

[6] J. Triesch and C. von der Malsburg. "A System for Person-Independent Hand Posture Recognition against Complex Background IEEE, 2001 Transactions on Pattern Analysis and

Machine Intelligence, Vol.23, N°. 12, December 2001.

[7] E. Sanchez-Nielsen, L. Antón-Canalis, M. Hernandez-Tejera, "Hand gesture recognition for human machine interaction", Journal of WSCG, Vol.12, No.1-3 (February 2003).

[8] C. Vogler and D. Metaxas. "A framework for recognizing the simultaneous aspects of American sign language". Computer Vision and Image Understanding, 81:358–384, 2001.

[9] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in Proc. 5th Annu. ACM Workshop Comput. Learning Theory, D. Haussler, Ed., 1992, pp. 144–152.

[10] C.-C. Chang, I.-Y Chen, and Y.-S. Huang, "Hand Pose Recognition Using Curvature Scale Space", IEEE International Conference on Pattern Recognition, 2002.

[11] A. Utsumi, T. Miyasato and F. Kishino, "Multi-Camera Hand Pose Recognition System Using Skeleton Image", IEEE International Workshop on Robot and Human Communication, pp. 219-224, 1995.

[12] Y. Aoki, S. Tanahashi, and J. Xu, "Sign Language Image Processing for Intelligent Communication by Communication Satellite", IEEE International Conf. On Acoustics, Speech, and Signal Processing, 1994.

[13] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff, "3D Hand Pose Reconstruction Using Specialized Mappings", IEEE International Con. on Computer Vision, pp. 378- 385, 2001.

[14] C. Tomasi, S. Petrov, and A. Sastry, "3D Classification + Interpolation", IEEE International Conf. on Computer Vision, 2003.

[15] W. T. Freeman and M. Roth, "Orientation Histograms for Hand Gesture Recognition", IEEE International Conf. on Automatic Face and Gesture Recognition, 1995.

[16] L. Bretzner, I. Laptev, and T. Lindberg, "Hand Gesture Recognition using Multi-Scale Color Features, Hierarchical Models and Particle Filtering", IEEE International Conf. on Automatic Face and Gesture Recognition, 2002.

[17] Jan A. Snyman (2005). Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms. Springer Publishing. ISBN 0-387-24348-8.

[18] Flecher, R., 1987. Practical Methods of Optimization, 2nd edition, New York: Wiley.

[19] Bertsekas, D.P., 1995. Nonlinear Programming, Belmont, MA: Athenas Scientific.

[20] SimonH., 1999. Neural networks: a comprehensive foundation, 2nd edition, McMaster University, Hamilton, Ontario, Canada, pp. 353-355.

[21] Kai-Bo Duan, S. Sathiya Keerthi, "Which Is the Best Multiclass SVM Method? An Empirical Study", Springer Berlin, 2005.

[22] Ron Meir, Nahum Shimkin, Shie Mannor, "Machine Learning Lecture Notes #7: Linear

Classification Methods", Electrical Engineering Dept., Technion – IIT, 2009.

[23] Aviad Barzilai, Adi Fuchs, Michael Kolomenkin, "Hand Gesture Classification Using Depth Information", CGM Laboratory undergraduate project, Electrical Engineering Dept., Technion – IIT, 2009

[24] Pearson, K. "On Lines and Planes of Closest Fit to Systems of Points in Space", Philosophical Magazine 2 (6): 559–572. 1901