Comparing Gesture Recognition Accuracy Using Color and Depth Information

Paul Doliotis, Alexandra Stefan, Christopher McMurrough, David Eckhard, and Vassilis Athitsos
Computer Science and Engineering Department, University of Texas at Arlington
Arlington, Texas, USA

ABSTRACT

In human-computer interaction applications, gesture recognition has the potential to provide a natural way of communication between humans and machines. The technology is becoming mature enough to be widely available to the public and real-world computer vision applications start to emerge. A typical example of this trend is the gaming industry and the launch of Microsoft's new camera: the Kinect. Other domains, where gesture recognition is needed, include but are not limited to: sign language recognition, virtual reality environments and smart homes. A key challenge for such real-world applications is that they need to operate in complex scenes with cluttered backgrounds, various moving objects and possibly challenging illumination conditions. In this paper we propose a method that accommodates such challenging conditions by detecting the hands using scene depth information from the Kinect. On top of our detector we employ a dynamic programming method for recognizing gestures, namely Dynamic Time Warping (DTW). Our method is translation and scale invariant which is a desirable property for many HCI systems. We have tested the performance of our approach on a digits recognition system. All experimental datasets include hand signed digits gestures but our framework can be generalized to recognize a wider range of gestures.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Input Devices and Strategies; I.4.8 [**Scene Analysis**]: Motion

General Terms

Experimentation

Keywords

gesture recognition, dynamic time warping, DTW, Kinect

1. INTRODUCTION

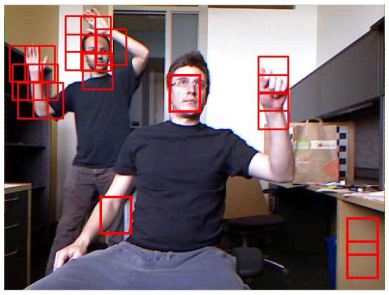

Figure 1: Detection of candidate hand regions based on skin color. Clearly, skin color is not sufficient to unambiguously detect the gesturing hand since the face, the non-gesturing hand, and other objects in the scene have similar color. On the other hand, for this particular scene, the gesturing hand is consistently among the top 15 candidates identified by skin detection.

Human-computer interaction interfaces need to be as intuitive and natural as possible. The user should ideally interact with machines without the need of cumbersome devices (such as colored markers or gloves [16]) or apparatus like remote controls, mouse and keyboards. Hand gestures can provide an alternative and easy means of communication with machines and could revolutionize the way we use technology in our daily activities. Successful applications of hand gesture systems can be found in various research and industry areas such as: game controlling [7], human-robot interaction [16], virtual environments, smart homes and sign language recognition [15], to name a few. Moreover with the advent and success of Microsoft's new camera, the Kinect, it has been clear that computer vision methods and specifically gesture recognition are becoming mature enough to be widely available to the public.

However, in order to create such successful and robust applications there is still much room for improvements. One key challenge for gesture recognition systems is that they must perform in uncontrolled real-world environments. This means heavily cluttered backgrounds with various moving objects and possibly harsh illumination conditions. Most hand gesture recognition systems assume that the gesturing hand can be reliably located in every frame of the input sequence. However, in many real life settings perfect hand detection is hard to achieve if not unrealistic. For example, in Figure 1 skin detection yields multiple hand candidates, and the top candidate is often not correct. Skin detection can be affected by varying or harsh illumination. Other visual cues commonly used for hand detection such as motion, edges, and background subtraction [4, 11] would also not perform well in backgrounds with moving objects which could be wrongly classified as moving hands.

We propose a method for building a robust hand detector that detects the gesturing hand in a scene by using motion detection based on frame differencing and depth segmentation. The depth values for each pixel in our images are given when we capture our data with a Kinect camera . Trajectories are created from detected hand locations at each frame of our test videos. In our database we have precomputed the trajectories for all our model digits. Finally we resort to a dynamic programming method, namely DTW [10], in order to compare the test and model trajectories and recognize the gestures. The main advantages of our method are:

- It performs very well even in very challenging environments with the presence of multiple "distractors" like moving objects, or skin colored objects (e.g., face, non-gesturing hand, background objects).
- It is robust to overlaps between the gesturing hand and the face or the other hand.
- It is translation and scale invariant; the gesture can occur in any part of the image.
- Unlike HMMs and CONDENSATION-based gesture recognition our method requires no knowledge of observation and transition densities, and therefore can be applied even if we have a single example per class.
- Our method can be generalized and applied to recognize a wider range of gestures, other than signs of digits.

We evaluate our technique on a vision-based digit recognition task. Each user can sign a digit ranging from 0 to 9 and our goal is to correctly classify the given digit. Similiar evaluation frameworks have been followed by other vision-based HCI systems (e.g., the virtual whiteboard by Black and Jepson [3], and the virtual drawing package by Isard [8]), to name a few).

2. RELATED WORK

Given a video sequence we can define a gesture as the trajectory of the points that correspond to the hand locations for each video frame. A gesture is therefore a sequence of 2D points or a time series. A popular method for comparing time series, that we employ in this paper, is the Dynamic Time Warping (DTW) [10]. We will describe in more detail the algorithm in section 5. DTW has been applied to successfully recognize a small vocabulary of gestures [5, 6].

A main disadvantage of DTW when used in gesture recognition is that it requires a perfect hand detector. For every frame we assume that we have the exact hand location. This assumption of course is hard to be satisfied in uncontrolled real-world environments, as mentioned in the previous section. To address this limitation we need to either build a really robust hand detector, like the one we propose in this paper, or we need to relax the assumption of a single candidate hand location per frame and allow for multiple detections of candidate hand regions. If we allow for more than one candidates, then we can employ Dynamic Space-Time Warping DSTW [1]. The dynamic space-time algorithm aligns a pair of query and model gestures in time, while at the same time it identifies the best hand location out of the multiple hypotheses available at each query frame.

Another similar approach is multiple hypothesis tracking (e.g., [12]) where multiple hypotheses are associated with multiple observations. The CONDENSATION-based framework can also be applied to gesture recognition [3]. Although in principle CONDENSATION can be used for both tracking and recognition, in [1] CONDENSATION was only used for the recognition part, once the trajectory had been reliably estimated using a color marker. Also, to use CONDENSATION we need to know the observation density and propagation density for each state of each class model, whereas in DSTW no such knowledge is necessary. Finally there is also a HMM framework proposed by Sato and Kobyashi [14]. In their method they extend the Viterbi algorithm so that multiple candidate observations can be accommodated at each query frame; the optimal state sequence is constrained to pass through the most likely candidate at every time step. However their approach is not translation invariant and it doesn't perform well in more challenging setting like the ones we use in our experiments.

3. OVERVIEW

Our method is an appearance based and example based gesture recognition method. In our experiments we define 10 classes representing the ten digits from 0 to 9, as shown in Figure 2 and 3.

Each digit can be formed by a gesture that has been signed by a user and is stored as a video sequence. For each digit class we have several training examples (videos) in our database. More specifically we have 10 different users performing 3 times each gesture digit, thus providing us 300 training examples. To make easier and automate the annotation for the training examples the user wears a colored glove and the background is fairly static and controlled. However we must stress that the same does not apply for the test sequences where the environment is far more challenging.

Given a test video sequence, that we must classify as one of our ten classes, we first need to find the hand location in each frame. In this paper we compare the performance we obtain using a color video, vs. the performance we obtain using a depth video, which provides, for every pixel, the depth. For the color videos, we use a hand detector which combines two visual cues, i.e., color and motion; both requiring only a few operations per pixel. Skin color detection is computationally efficient, since it involves only a histogram lookup per pixel. Similarly, motion detection, which is based on frame differencing, involves a small number of operations per pixel.

For the depth video, we use a hand detector based on motion from frame differencing which we combine with a depth segmentation according to the depth information we have for each pixel. Our detector can return a single best hand location that will be the input for DTW.

Recognizing the input gesture is done using the nearest neighbor classification framework. The similarity measure that we use for the 1NN scheme is the score returned by the DTW. The DTW algorithm temporally aligns two sequences, a query sequence and a model sequence, and computes a matching score, which is used for classifying the query sequence. The time complexity of the basic DTW algorithm is quadratic in the sequence length, but more efficient variants have been proposed [13, 10]). In DTW, it is assumed that a feature vector can be reliably extracted from each query frame. In the following sections we describe in detail each module of our method.

4. DETECTION AND NORMALIZATION

4.1 Detection

The first hand detector that we used in our experiments is applied on RGB color images and is based on motion and skin detection. To build the skin color model we use a generic skin color histogram [9] to compute a skin score image, in which each pixel in the frame gets assigned a value denoting the probability of being skin. The

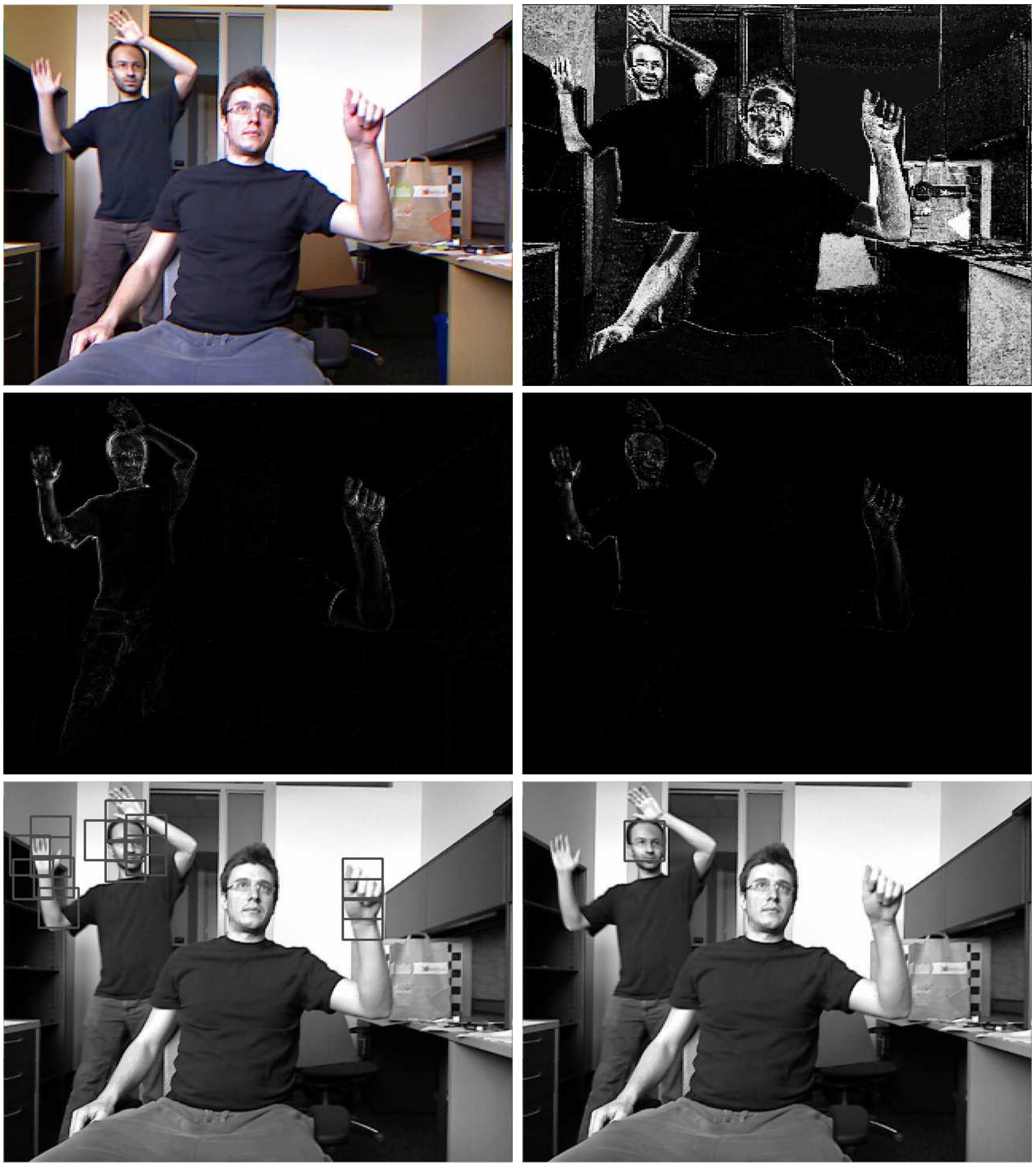

Figure 4: **Hand detection in color images: original image (top left), skin detection (top right), frame differencing (middle left), multiplication of skin and frame differencing scores (middle right), top 15 hand candidates (bottom left), single top candidate (bottom right). We see that the hand detector here fails to detect the hand of the gesturing person.**

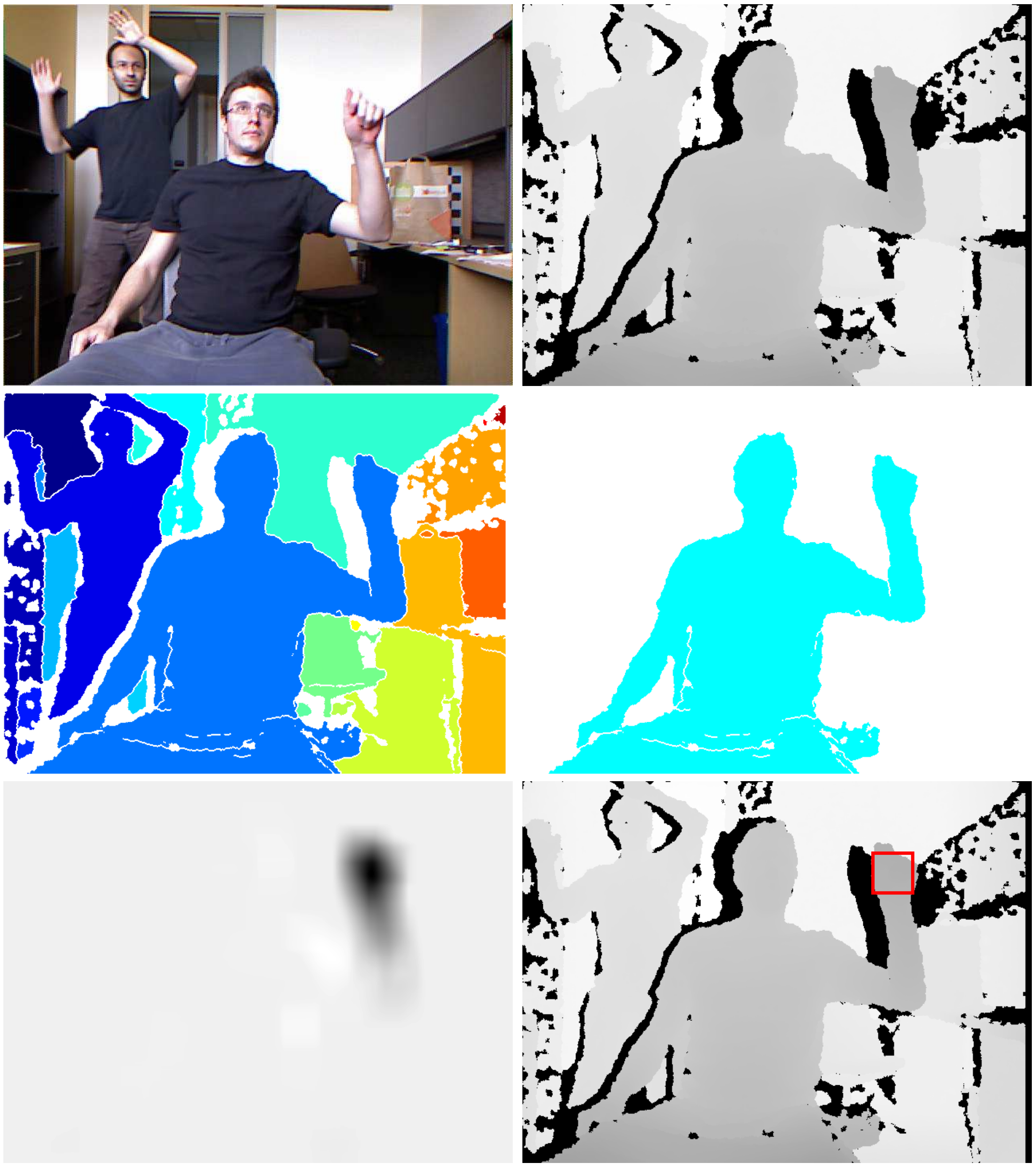

Figure 5: **Hand detection in depth images: original image (top left), depth image (top right), segmentation using depth (middle left), the connected component corresponding to the gesturing human (middle right), scores based on multiplication of frame differencing and depth (bottom left), single top candidate (bottom right). We see that the depth detector here successfully detects the hand of the gesturing person.**

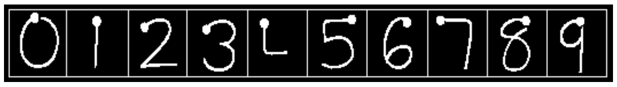

Figure 2: **Palm's Graffiti digits.**

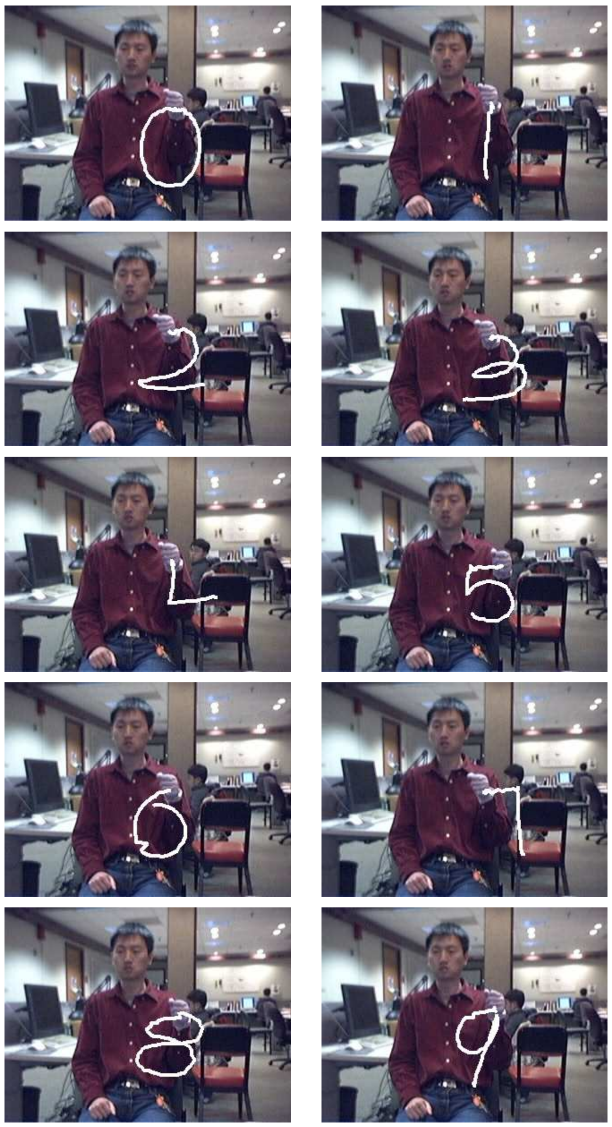

Figure 3: **Example model digits extracted using a colored glove. We reuse the figure and actual videos from [1].**

motion detector computes a score matrix with the same size as the original image by using frame differencing (frame differencing is the operation of computing, for every pixel, the minimum of two values; the absolute value of the difference in intensity between the current frame and the previous frame and the absolute value of the difference in intensity between the current and the next frame). Then we multiply element by element the motion score matrix with the skin score image to obtain the hand score image. Next we compute for every subwindow of some predetermined size the sum of pixel scores in that subwindow. Then we extract the subwindow with the highest sum. The gesturing hand is typically covered by one or more of these subwindows (See Figure 1). Figure 4 illustrates examples of input, output, and intermediate steps for this detector.

Each pixel in an image taken from a Kinect camera is assigned a value according to how close or far it is from the plane defined by the camera lenses. For depth video, our proposed detector is based on motion and depth information and is applied on depth images captured from a Kinect camera. The depth images are grayscale images with values ranging from [0-2046] and value 2047 denotes an invalid depth pixel. First we find the connected component of the person signing the digit. To achieve this we calculate for every pixel the absolute value of the difference, in depth, between that pixel and the top, bottom, left and right neighboring pixels. If the absolute value is over a threshold (10 in our experiments) the pixel is zero otherwise one. In this way we create a binary image where edges are depicted on areas with abrupt changes in depth intensity. Now we have a cleanly segmented image compromising of connected components that have the following property: within each component the depth intensity values of neighboring pixels never increase over 10. The next step is to calculate the average depth for the 5 biggest connected components. The component with the lowest mean depth value is assumed to be the person signing the digit. Now that we know which pixels belong to the person we calculate their max and median depth values.

Next, we calculate a score matrix based on frame differencing, and another score matrix which is our depth image minus the median depth value for the person. We multiply element by element those two matrices and we apply a mask over the final matrix. The mask has the following properties:

- all invalid pixels in the previous, current and next frame are zero. The reason we do this is because the captured depth images have a lot of noise which can be regarded as ”motion”.
- all pixels in the motion score matrix that have a value lower than 1 are zero
- all pixels that their intensity value is over the max depth value are zero.

After we have applied the mask on our score matrix we use a vertical 1D filter of ones with some predefined size and finally on the new matrix we apply the same filter horizontally. The min value of our final score matrix denotes where our hand is located. Figure 5 illustrates examples of input, output, and intermediate steps for this detector.

4.2 Normalization

When the hand detector is applied, we can segment the hand region. Then we extract our features that will be given as inputs to DTW. We use a basic feature which is the 2D position (x, y) of the segmented region centroid. So in the end, for every hand region within a frame a 2D feature vector $Q = (x, y)$ is extracted. Cleary, position is translation and scale depended and we must use a normalization procedure before we give our sequences of 2D vectors as input to DTW.

Normalization is achieved in the following way: After we have the set of all 2D points corresponding to hand locations, we can calculate the Minimum Enclosing Circle (MEC) of that set. Then we find the bounding square box that has the same center as the MEC and its width length is twice the circle's radius. Then we resize the square to a fixed size of 300x300. By normalizing our features our system becomes translation and scale invariant and increases our recognition rates.

5. DYNAMIC TIME WARPING

One of several publications that describe the DTW algorithm is [10]. In this section we briefly describe the algorithm presented in that paper.

Let $M = (M_1, \ldots, M_m)$ be a model sequence in which each M_i is a feature vector and let $Q = (Q_1, \ldots, Q_n)$ be a query sequence in which each Q_j is another feature vector.

A warping path W defines an alignment between M and Q. Formally, $W = w_1, \ldots, w_T$, where $\max(m, n) \leq T \leq m + n - 1$. Each $w_t = (i, j)$ specifies that feature vector M_i of the model is matched with feature vector Q_j. The warping path is typically subject to several constraints:

- **Boundary conditions:** $w_1 = (1, 1)$ and $w_T = (m, n)$. This requires the warping path to start by matching the first frame of the model with the first frame of the query, and end by matching the last frame of the model with the last frame of the query.

- **Temporal continuity:** Given $w_t = (a, b)$ then $w_{t-1} = (a', b')$, where $a - a' \leq 1$ and $b - b' \leq 1$. This restricts the allowable steps in the warping path to adjacent cells along the two temporal dimensions.

- **Temporal monotonicity:** Given $w_t = (a, b)$ then $w_{t-1} = (a', b')$ where $a - a' \geq 0$ and $b - b' \geq 0$. This forces the warping path sequence to increase monotonically in the two temporal dimensions.

There are exponentially many warping paths that satisfy the above conditions. however we are only interested in the path that minimizes the warping cost:

$$DTW(Q, C) = \min_{w_1, \ldots, w_T} \{ \sqrt[2]{\sum_{t=1}^{T} w_k} \} \qquad (1)$$

This path can be found using dynamic programming to evaluate the following recurrence, which defines the cumulative distance $\gamma(i, j)$ as the distance $d(i, j)$ found in the current cell and the minimum of the cumulative distances of the adjacent elements:

$$\gamma(i, j) = d(q_i, c_j) + min\{\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\} \qquad (2)$$

The Euclidean distance between two sequences can be seen as a special case of DTW where the kth element of W is constrained such that $w_k = (i, j)_k, i = j = k$. Note that it is only defined in the special case where the two sequences have the same length. The time and space complexity of DTW is $O(nm)$.

6. EXPERIMENTS

To evaluate the performance of our method we have created a hand-signed digit recognition system. The training videos that we use are publicly available, as described in [2]. In that data, the trajectories can be easily found, since the persons signing the digits are wearing a green colored glove. This is a convention that we allow only on our training data which we then preprocess offline in order to create fine trajectories. However this is not the case for the testing data. Our test video sequences have been captured in some really challenging settings so as to measure the robustness of our proposed and previous methods. The test data have been collected with the Kinect camera using an image resolution of 480x640. In more detail our datasets have been organized as follows:

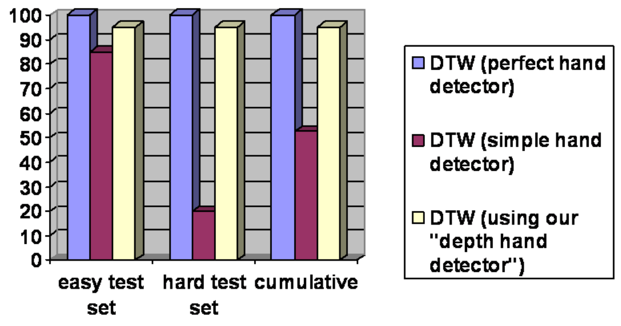

Figure 6: results

- Training examples: 300 digit exemplars (30 per class) were stored in the database (See Figure 3). A total number of 10 users have been employed to collect all training data

- Test gestures: 40 digit exemplars (20 per class) were used as queries. For 20 of them the users were wearing short-sleeve shirts contrary to most gesture recognition methods. We will refer to this half of the test set as the *easy test set*. For the remaining 20 exemplars, we have created even more challenging conditions, with people and various objects moving constantly in the background. In this way we want to demonstrate that most of the previous methods fail while our proposed method remains robust even in the harshest conditions. We will refer to this half of the test set as the *hard test set*. A total number of 2 users have been employed to collect all test data.

It is important to note that recognition was performed in a *user-independent* manner: we never use a model video sequence as a test query. The test videos have been collected from users that do not appear in the training videos. Using a hand detector we extract a trajectory from a test query. Then we compare that trajectory with all the pre-computed training trajectories and using 1-NN Nearest Neighbor Classification we classify our test signed digit as one of the 10 classes, ranging from 0 to 9.

First we test the performance of DTW given that we have a perfect hand detector for our test sequences. To extract the trajectories in this case we have resorted to manual annotation. Naturally, the gesture recognition is 100% accurate and all digits are classified correctly. Then we test DTW by employing another detector which uses motion and skin color to locate the hand in each frame. This will also serve as a baseline performance for our system. For the *easy test set* we achieve an accuracy of 85% while for the *hard test set* the accuracy reduces drastically down to 20% emphasizing the weakness of previous methods in such challenging environments. Finally, our method achieves an accuracy of 95% for both test sets. All results are depicted in Figure 6.

7. DISCUSSION AND FUTURE WORK

We have presented a translation and scale invariant gesture recognition method that can achieve high performance even in challenging environments with backgrounds full of moving people and objects. Our contribution is a hand detector based on motion detection and depth segmentation that can accurately locate the hand at each frame. On top of our detector we normalize our features and then we employ the Dynamic Time Warping algorithm to recognize the gestures. Incorporating translation and scale invariance makes our system more user friendly since the user has fewer restrictions as to where exactly he needs to be placed with respect to the camera. However some restrictions remain and one of them is viewpoint invariance. In our method we assume that the user always faces the camera so that we have a frontal view of the gesture. Future work will be to implement a 3D human articulated motion tracker. If we know the 3D motion of the user we can easily convert to a frontal view representation.

Another open problem that needs investigation is temporal segmentation. In our approach we manually annotate the begining and end frame for each gesture. Automating this procedure (e.g. by using a distinct pose for the non-gesturing hand) remains future work.

Finally, we would like to expand our gesture recognition system in order to accomodate more challenging gestures from other domains such as the American Sign Language.

8. REFERENCES

[1] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff. Simultaneous localization and recognition of dynamic hand gestures. In *IEEE Motion Workshop*, pages 254–260, 2005.

[2] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(9):1685–1699, 2009.

[3] M. Black and A. Jepson. Recognizing temporal trajectories using the condensation algorithm. In *IEEE International Conference on Automatic Face and Gesture Recognition(AFGR)*, pages 16–21, 1998.

[4] F. Chen, C. Fu, and C. Huang. Hand gesture recognition using a real-time tracking method and Hidden Markov Models. *Image and Video Computing*, 21(8):745–758, August 2003.

[5] A. Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *Recognition, Analysis and Tracking of Faces and Gestures in Real-time Systems (RATFG-RTS)*, pages 82–89, 2001.

[6] T. Darrell and A. Pentland. Space-time gestures. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 335–340, 1993.

[7] W. Freeman. Computer vision for television and games. In *Recognition, Analysis and Tracking of Faces and Gestures in Real-time Systems (RATFG-RTS)*, page 118, 1999.

[8] M. Isard and A. Blake. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. In *European Conference on Computer Vision (ECCV)*, pages 893–908, 1998.

[9] M. Jones and J. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision (IJCV)*, 46(1):81–96, January 2002.

[10] E. Keogh. Exact indexing of dynamic time warping. In *International Conference on Very Large Databases (VLDB)*, pages 406–417, 2002.

[11] J. Martin, V. Devin, and J. Crowley. Active hand tracking. In *Automatic Face and Gesture Recognition*, pages 573–578, 1998.

[12] C. Rasmussen and G. Hager. Probabilistic data association methods for tracking complex visual objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(6):560–576, June 2001.

[13] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. In *IEEE Transactions on Acoustics, Speech, and Signal Processing*, volume 34(1), pages 43–49, 1978.

[14] Y. Sato and T. Kobayashi. Extension of hidden markov models to deal with multiple candidates of observations and its application to mobile-robot-oriented gesture recognition. In *International Conference on Pattern Recognition (ICPR)*, volume 2, pages 515–519, 2002.

[15] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. In *IEEE International Symposium on Computer Vision*, pages 265–270, 1995.

[16] J. Triesch and C. von der Malsburg. Robotic gesture recognition. In *Gesture Workshop*, pages 233–244, 1997.