Real-time gesture recognition using deterministic boosting

Raymond Lockton and Andrew W. Fitzgibbon
Department of Engineering Science
University of Oxford.

Abstract

A gesture recognition system which can reliably recognize single-hand gestures in real time on a 600Mhz notebook computer is described. The system has a vocabulary of 46 gestures including the American sign language letterspelling alphabet and digits. It includes mouse movements such as drag and drop, and is demonstrated controlling a windowed operating system, editing a document and performing file-system operations with extremely low error rates over long time periods.

Real-time performance is provided by a novel combination of exemplar-based classification and a new "deterministic boosting" algorithm which can allow for fast online retraining. Importantly, each frame of video is processed independently: no temporal Markov model is used to constrain gesture identity, and the search region is the entire image. This places stringent requirements on the accuracy and speed of recognition, which are met by our proposed architecture.

1 Introduction

Gesture recognition is an area of active current research in computer vision. The prospect of a user-interface in which natural gestures can be used to enhance human-computer interaction brings visions of more accessible computer systems, and ultimately of higher bandwidth interactions than will be possible using keyboard and mouse alone.

This paper describes a system for automatic real-time control of a windowed operating system entirely based on one-handed gesture recognition. Using a computer-mounted video camera as the sensor, the system can reliably interpret a 46-element gesture set at 15 frames per second on a 600MHz notebook PC. The gesture set, shown in figure 1, comprises the 36 letters and digits of the American Sign Language fingerspelling alphabet [9], three 'mouse buttons', and some ancillary gestures. The accompanying video, and figure 4, show a transcript of about five minutes of system operation in which files are created, renamed, moved, and edited—entirely under gesture control. This corresponds to a per-image recognition rate of over 99%, which exceeds any reported system to date, whether or not real-time. This significant improvement in performance is the outcome of three factors:

1. The *general engineering* of the system means that preprocessing is reliable on every frame. Lighting is controlled with just enough care to ensure that most skin pixels are detected using simple image processing. The user wears a coloured wrist band which allows the orientation of the hand to be easily computed.

2. An *exemplar-based classifier*[11, 19] ensures that recognition of the gesture label from a preprocessed image uses a rich, informative model, allowing a large gesture vocabulary to be employed.

BMVC 2002 doi:10.5244/C.16.80

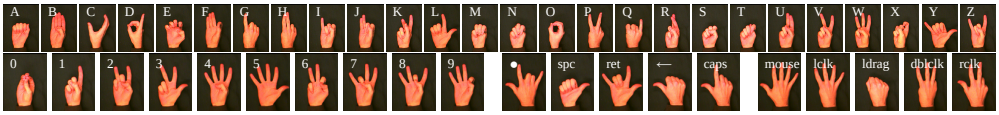

Figure 1: The gesture set to be recognized. The letters and digits of American sign language have been modified as follows: the dynamic gestures 'J' and 'Z' are replaced with static versions which can be recognized in a single frame; the digits whose sign is identical to a letter—and which therefore are distinguished by human signers based on context—have been modified to be distinguishable without such context; and ten additional gestures have been added for operations such as 'delete', 'enter' and mouse button actions.

3. The basic exemplar-based recognition is significantly sped up both by conventional exemplar clustering, and by a novel variant of the pattern-recognition technique of *boosting*[10], yielding orders of magnitude speedups over conventional implementations.

A useful analogy for the strategy employed by the system is brute-force matching of each input image against a database of template images for each stored gesture. The novel contribution of the paper is in the extension of two emerging strands of research to reduce the enormous complexity of such approaches so that real-time implementation is possible. The algorithm reduces the computational cost from $O(10^8)$ pixel operations per frame to about $O(10^5)$, a speed improvement of three orders of magnitude (or from one minute per frame to 15 frames per second). Because this is achieved with a negligible loss of accuracy, the system remains almost as accurate as a full template-matcher would be. The result is the first system of which we are aware to combine the power of exemplar-based methods with the efficiency of boosting in order to build a large-vocabulary real-time recognition system.

The rest of the paper describes the design and implementation of this system. In order to clearly define the problem to be solved, the next section briefly describes the image capture system, and the preprocessing which segments skin pixels and normalizes for rotation and translation. Armed with the notation from that introduction, section 3 situates our work in the existing research on gesture recognition systems, and compares exemplar-based and parametric model-based approaches to recognition. This is followed by a short description of the boosting paradigm for generating efficient, high-reliability statistical classifiers. Section 4 details the construction and implementation of our system, and section 5 shows the results of experimental evaluation of the system, the demo transcript, and concludes the paper.

2 Acquisition and preprocessing, problem statement

Gestures are acquired using a desktop camera observing a 40×40 cm^2 workspace, under room lighting. Skin pixels are detected as those whose colour is inside an axis-aligned box in RGB colour space. This is reliable for most pixels in a typical hand image, but results in lost pixels at the edges of the hand, on highlights, and in shadowed internal areas. However, the simplicity of the technique means that it is readily implemented in real time on a notebook computer, with time to spare for the recognition stage. The variation due to misclassification of skin pixels becomes a minor addition to the within-class variation dealt with in the later recognition stage. The user wears a wristband in order to allow hand orientation and scale to be computed robustly. A certain amount of engineering has gone into the reliable separation of hand and wristband pixels, which is not described here, but details are in an associated

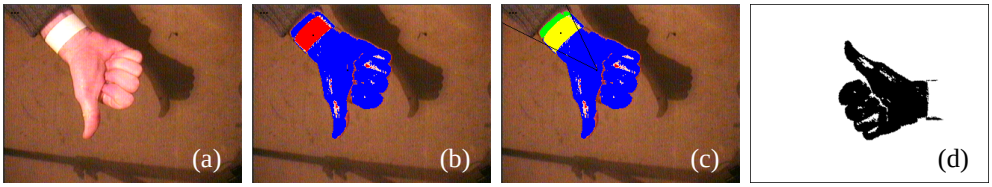

Figure 2: Preprocessing stages. Webcam images (a) are thresholded (b) in RGB for speed. Detected wristband and hand centroids (c) are used to transform to canonical frame (d). Note that in the final application, the canonical frame image is never explicitly formed—the small subset of pixels queried is pulled directly from the input image. Note also that several pixels on the hand have not been classified as skin: the recognition stage will learn to ignore these.

technical report [2]. Figure 2 illustrates the procedure for a typical input image.

For the remainder of the paper, images will be represented as a 1D column vectors $\mathbf{x} \in \mathbb{R}^N$ which are the concatenation of the image columns. All images will be considered to be the binarized, canonicalized versions as in figure 2d. We wish to recognise gestures based on a previously acquired training set. Let us assume that we have K gestures, and for the k^{th} gesture we have obtained M_k training examples. Throughout the rest of the discussion, we will assume without loss of generality that all gestures have the same number of training images M, so $M_k = M\ \forall k$. In our application, $K = 46$ and $M = 100$, so there are a total of 4600 training examples; N is the number of image pixels, 320×240 here.

The training examples are denoted $\mathbf{x}_{mk}$ for $m = 1 \dots M$ and $k = 1 \dots K$. By convention, the correct label for the example $\mathbf{x}_{mk}$ is k. A *classifier* is a function $c(\mathbf{x}) = l$ which takes a test image $\mathbf{x}$ and returns a label $l \in \{1 \dots K\}$. A classifier which explains the training data perfectly will obey

$$c(\mathbf{x}_{mk}) = k \quad \forall m, k$$

We trust in ingenuity and a representative training set to find classifiers for which good performance on the training data implies good performance on test examples. The task of this paper is to define an accurate classifier which may be computed rapidly enough to allow real time operation.

A *soft classifier* does not return a label, but instead assigns a likelihood to a given (image, label) combination: $c(\mathbf{x}, l) : \mathbb{R}^N \times \{1 \dots K\} \mapsto [0, 1]$ A likelihood close to one implies that the image is likely to be that label, a likelihood close to zero says that the assignment is unlikely. A perfect soft classifier will assign ones on the training set for the correct label, and zeros otherwise, i.e. $c(\mathbf{x}_{mk}, l) = \delta_{lk}\ \forall l, m, k$ where δ is the Kronecker delta. We note also that the soft classifier is not required to return probabilities. In particular, we do not require that $\sum_{l=1}^{K} c(\mathbf{x}, l) = 1$.

3 Background

This paper draws from a few strands of research: gesture recognition, exemplar-based tracking, and pattern classification. This section discusses the related literature, and emphasizes the areas in which this paper innovates.

Gesture recognition: Much work has been done on gesture recognition over the years, and this section does not attempt a full literature review, but rather points to some prototypical

systems. See [7, 14] for reviews.

Starner et al [17] describe a system which demonstrates impressive results by combining an extremely spartan representation of shape (sixteen measurements based on moments of inertia of the region within the silhouette) with a hidden Markov model of sign transitions. The system can recognize in real time, but has a restricted word-based vocabulary and is strongly driven by the Markov model. The implications of this are discussed further below.

Bowden and Sarhadi [5] use a nonlinear point distribution model, allowing a much richer description of hand shape, and augment this with a Markov model describing the transition probabilities of English. However, they note that the Markov model is important for the success of their technique, and processing is not claimed or expected to be real time. In addition, training and initialization of PDMs remains difficult to achieve consistently. Part of the contribution of our work is to show that exemplar-based techniques can perform as well as PDMs on a real-world problem.

The appearance-based approach of [3, 13] computes a PCA of the canonicalized hand images, and may be viewed as the parametric model-based analogue of this paper's exemplar approach. In general, however, the PCA will require nonlinear extensions before the gesture set size can be expanded to the size used in this paper.

Other work on gesture recognition using multiple cameras [6, 15] or 3D hand tracking [1, 12, 18] is of relevance to this work, but has not yet been demonstrated to cope in real time with the large vocabulary and complex and rapid gesture changes which signing involves. Even with 3D information, the problem of classification and recognition of the gestures remains. We would hope that some of the strategies described in this paper would also be useful with 3D trackers.

Markov models: Most current sign-language recognition depends on the representation of temporal constraints via Markov models [5, 6, 17, 21] to achieve high-accuracy operation. The difficulty with systems based on Markov models of temporal gesture behaviour is that they restrict the range of gestures that can be accurately recognized. For example, in the file-handling application demonstrated here, the temporal statistics of input gestures do not always follow those of ASL or the English language. File names may be abbreviated, notes may be made in other languages. To deal with these situations, a gesture recognizer which has high accuracy on individual frames of video without temporal constraints is needed. Furthermore, we wish to allow the hand to be removed from the workspace or occluded without any loss in accuracy or any pause for re-initialization, so tracking based on spatio-temporal coherence cannot be the means by which we claim real-time performance. This in turn places stringent requirements on the accuracy of the raw recognition engine. Of course, Markov models could be readily added to the engine proposed here, which would be expected to increase performance.

Exemplars versus parametric models: In this work, within-class variation is due to shadowing, 3D positioning, and the user's tendency to form the same gesture in slightly different shapes each time. A key assumption is that such variation is best encapsulated by learning from training examples. Two important paradigms are (a) the learning of parametric models and (b) the recent emergence of exemplar-based strategies. Parametric models are exemplified by point-distribution models [8] and their nonlinear extensions [4, 16], and produce models which live in a vector space, meaning that (at least locally) linear combinations of the model parameter vectors produce new examples of the learnt model. In contrast, exemplar-based approaches [11, 19] relax the vector space to be a metric space: there exists a distance metric which can compare two models, but no rule is provided for the generation of new examples.

Comparing the two strategies, some general observations can be made: it is often easy to provide a distance metric, but hard to build a parametric model. On the other hand, parametric models can generalize from small amounts of training data. For parametric models, the set of parameters which generate physically valid models may be difficult to characterize: given a PDM with sufficient variability to model the full set of gestures in figure 1, one would expect to find that the set of parameters which generate valid gestures live in a complex nonlinear subspace of the linear vector space. The final difficulty lies in automatic initialization and training of these techniques, which remains an open research problem. Exemplars, on the other hand, are trivially trained and at runtime the templates are easy to extract from the image stream, but the techniques require large training sets, and initially required significant computational effort. Gavrila and Philomin's hierarchical matching [11] has made one step towards real time recognition from large databases, the deterministic boosting algorithm introduced in this work is another. In summary, exemplars allow the construction of reliable and robust systems, and can now be made fast as well.

Boosting: Boosting [10] is one of a class of techniques which allows the combination of several statistical classifiers in order to generate a consensus classifier which attains high reliability and accuracy. The component *weak classifiers* are typically fast, but low quality, classifying only slightly better than at random for two-class problems. More correctly, "boosting" refers to one of Freund and Schapire's *AdaBoost* algorithms [10] for training such combined systems and selection of the set of weak classifiers.

The basic idea of boosting is simple. We have a classification problem, and access to a weak classifier—say for example, a neural network. Most importantly, we have a way to tell the weak classifier to concentrate on getting certain examples in the training set right. We train the weak classifier on our training data, favouring no particular examples. As expected, the result is a classifier, call it c_1, which explains the training data, but whose performance is maybe only a bit better than random. The key step follows: the weak classifier is retrained, concentrating on getting the "hard" examples right. The original classifier is not discarded; a combined classifier is built which is a weighted average of the results of c_1 and the newly trained classifier, c_2. Now, the combined classifier has a new set of "hard" examples, hopefully smaller than either of the "hard sets" of c_1 and c_2. Continuing the process can be shown to consistently yield a high-accuracy classifier providing the weak classifier's performance is better than random. For details of how the classifier performance on the training set is converted to favour hard examples, and of how the weak classifiers are combined, the reader is referred to [10]. The well-known disadvantage of boosting is that as the hard sets get harder, the likelihood of finding a good classifier drops, so that training is notoriously slow.

In a recent computer vision application of boosting, Viola et al [20] combined very simple face detectors to build a real-time face detection system with excellent accuracy. The simple sensors are based on thresholding of Haar wavelet responses, which may be computed extremely quickly. In this paper the sensors used are even simpler: a single pixel's skin/non-skin status is all that is queried. This paper's new *deterministic boosting* algorithm speeds up the boosting of extremely weak classifiers such as these.

4 The algorithm

In order to outline the algorithm used in this paper, we first describe it in terms of nearest-neighbour template matching, and then describe the techniques which render it computation-

ally feasible. In nearest-neighbour matching, a new binary image $\mathbf{x}$ is compared against all the training examples, and the label of the closest example is reported. Affinity between two binary images $\mathbf{x}$ and $\mathbf{y}$ is a centered correlation: $a(\mathbf{x},\mathbf{y}) = \frac{4}{N}\sum_{n=1}^{N}(x_n - \frac{1}{2})(y_n - \frac{1}{2})$, whose value is $+1$ when $\mathbf{x}$ and $\mathbf{y}$ are identical and reaches a minimum of -1 when $\mathbf{y} = 1 - \mathbf{x}$. We may define a distance measure as $d(\mathbf{x},\mathbf{y}) = 1 - a(\mathbf{x},\mathbf{y})$. The nearest-neighbour classifier c_{NN} is

$$c_{NN}(\mathbf{x}) = \underset{k \in \text{gestures}}{\operatorname{argmin}} \left(\min_{m \in \text{examples}} d(\mathbf{x}, \mathbf{x}_{mk}) \right)$$

and its soft-classifier form is $c_{NN}(\mathbf{x}, l) = \max_m \frac{1}{2} a(\mathbf{x}, \mathbf{x}_{ml}) + \frac{1}{2}$. The computational complexity of template matching is $O(MKN)$—linear in the number of images, gestures and pixels.

4.1 Exemplar clustering

To reduce the computational burden, the first strategy is to cluster the training examples [11, 19]. We wish to choose a subset of the training images for each gesture such that nearest-neighbour classification in the subset produces results as close as possible to the full NN classifier. To this end, we follow Toyama and Blake [19], and implement a medoid-based clustering algorithm. Each gesture is processed separately, so the task for gesture k is to take the set of training images $\mathcal{T}_k = \{\mathbf{x}_{1,k} \ldots \mathbf{x}_{M,k}\}$ and replace it with a subset $\mathcal{C}_k = \{\mathbf{x}_{m_1,k} \ldots \mathbf{x}_{m_{r_k},k}\}$ for a reduced number of examples r_k. We avoid algorithms such as k-medoids which require that we predict in advance the desired number of clusters. Instead, we choose a group of cluster centres and ensure that each of the original training examples is within a threshold distance of at least one cluster centre. Formally, we choose cluster centres $\mathcal{C}_k \subset \mathcal{T}_k$ such that

$$\forall \mathbf{x}_{mk} \in \mathcal{T}_k, \;\; \exists \mathbf{y} \in \mathcal{C}_k \text{ such that } a(\mathbf{x}_{mk}, \mathbf{y}) > \alpha$$

Although choosing a subset $\mathcal{C}_k$ which satisfies this threshold and has the smallest possible number of elements is an NP-hard problem, we have found that a greedy algorithm[1] provides adequate results.

The algorithm was applied to a set of 100 examples of each of 46 gestures. The threshold on minimum affinity α was set to 0.95. The numbers of exemplars for the gestures 'A' through 'E' were reduced from 100 each to $1, 1, 3, 4, 8$ respectively and the total for all 46 gestures was reduced from 4600 to 183.

Finally, the cluster contents are summarized for each exemplar by a single **coherence map**, defined as follows. Each input example is assigned to the cluster with whose centre it has highest affinity. This defines a set of assigned images $\mathcal{S}_{ik}$ for each cluster centre $\mathbf{x}_{m_i,k}$ in $\mathcal{C}_k$. The coherence map $\mathbf{v}_{m_i,k}$ is just the pixelwise mean of each cluster $\mathbf{v}_{m_i,k} = \frac{1}{\#\mathcal{S}_{ik}} \sum_{\mathbf{x} \in \mathcal{S}_{ik}} \mathbf{x}$. Each coherence map encodes, for each pixel, the number of times that pixel was detected as skin over the training images in the cluster. Specifically, for a coherence map $\mathbf{v}$, we have $v_n = 1$ if pixel n was skin in every image in the cluster, $v_n = 0$ if it was always background, and intermediate values if the pixel was detected as both, due to within-class variation. Figure 3 shows coherence maps for one cluster of each of the gestures 'M' and 'N'. The nearest neighbour classifier c_{NN} is defined exactly as before, albeit with real-valued rather than binary exemplars. This phase reduced recognition accuracy on the training set from 100%, as

[1] Set $\mathcal{C}_k = \{\}$. While $\mathcal{T}_k$ is not empty: (add head($\mathcal{T}_k$) to $\mathcal{C}_k$, remove from $\mathcal{T}_k$ all $\mathbf{y}$ for which $a(\mathbf{y},\mathbf{x}) > \alpha$)

(m) (n) (pseudo-loss $\mathbf{p}$) (boosted $\mathcal{N}$)

Figure 3: **Left** (m,n): Within-class variation map $\mathbf{v}$ for each of two gestures. Black pixels are detected as skin in each training image, white pixels are always background, and gray pixels are those whose classification varied through the training set. **Middle** (pseudo-loss): Pixels which distinguish m and n. These are *reliable* (black or white) for both gestures, and *distinctive* (black for m, white for n or vice versa). This is a map of $\mathbf{p}$ for this 2-gesture set (§4.2). **Right** (boosted): The final set $\mathcal{N}$ of 1199 pixels used for classification over the 46 gestures. Notice that pixels near the wrist are ignored—although reliably segmented as skin, they are not distinctive as all gestures share them.

is always achieved on the training set with nearest-neighbour classifiers, to about 99.5%, and speed is improved by a factor of 20. The training set is now represented by a set of coherence maps $\mathbf{v}_i$ for i in $1 \ldots M' = \sum_{k=1}^{K} r_k$ and an associated label assignment k_i which indicates the gesture to which $\mathbf{v}_i$ corresponds.

4.2 From templates to per-pixel sensors

The next speedup is obtained by replacing the nearest-neighbour classifier with a collection of much weaker classifiers, and combining these classifiers to optimize their performance on the training set. The weak classifiers used in this paper embody wholeheartedly their epithet—to decide the gesture represented by an image $\mathbf{x}$, a single pixel is queried and compared to the training set. At best, the pixel, pixel n say, will be skin ($x_n = 1$) for all training examples of some gestures and background ($x_n = 0$) for all examples of the remainder. At very best, it will be skin for half the gestures and background for the other half. If this very best applied to several different pixels, one could imagine a tree-like recognition strategy, in which each examined pixel splits the number of candidates in half, and only six pixels would need to be queried to distinguish 64 gestures. It is towards this sort of speedup that we wish to work, although in reality such a scheme would be far from robust. Even if six such pixels were found to correctly classify the training set, a single segmentation error would give an erroneous classification with full confidence. However, one might hope that a few hundred pixels could be found, from which a consensus classification could be extracted, generating a reliable, fast classifier.

The per-pixel classifier which we use is a soft classifier. Given a set of training exemplars, represented as coherence maps and labels $(\mathbf{v}_i, k_i)$, a test image $\mathbf{x}$ is classified based only on the value of pixel n:

$$c_n(\mathbf{x}, l) = \max_{i \text{ such that } k_i = l} \begin{cases} [\mathbf{v}_i]_n & \text{if } x_n = 1 \\ 1 - [\mathbf{v}_i]_n & \text{if } x_n = 0 \end{cases}$$

Given N' such classifiers, corresponding to pixels $\mathcal{N} = \{n_1 \ldots n_{N'}\}$, the combined classifier cc is obtained by averaging the weak classifier results: $cc(\mathbf{x}, l) = \frac{1}{N'} \sum_{n \in \mathcal{N}} c_n(\mathbf{x}, i)$. There-

fore, after training in order to find $\mathcal{N}$, we obtain the combined recognition algorithm R, which assigns a gesture label to a canonical-frame image $\mathbf{x}$ as follows: $R(\mathbf{x}) = \mathrm{argmax}_k\, cc(\mathbf{x}, k)$.

The remaining detail is how to choose the set of query pixels $\mathcal{N}$ from the set of all pixels. One option would be simple random sampling, another would be to use AdaBoost. However, in this work we can do better than either alternative. We define a quality function for each weak classifier, analogous to AdaBoost's pseudo-loss function. The quality function for c_n has a high value if pixel n is a good discriminant across the training set, and if it splits the dataset well. Thus if we take a certain pixel n, and consider the set of coherence-map values at that pixel: $\mathcal{V} = \{[\mathbf{v}_1]_n, \ldots, [\mathbf{v}_{M'}]_n\}$ we wish to minimize the following pseudo-loss function

$$p_n = \left|\sum_{v\in\mathcal{V}} (v - 0.5)\right| - \sum_{v\in\mathcal{V}} |v - 0.5|$$

The image $\mathbf{p}$ which is the reassembly of all the p_n on a two-gesture set is shown in figure 3.

We may now *deterministically* select the weak classifiers: sorting the set of all pixels on p_n yields an ordered set P of the single-pixel classifiers from most to least effective at classifying the training set. Choosing the first few hundred of these would be expected to yield good recognition performance at significantly lower computational cost than matching over all pixels. However, one final refinement is required. We wish each of our single-pixel classifiers to classify the gesture set into different partitions. Thus, we cluster the classifiers into subsets which distinguish the same gestures. In order to perform the classification, we require a metric which measures whether two pixel classifiers perform the same job. Again, we can derive an efficient function to compute this metric: $D(n_1, n_2) = \sum_i |[\mathbf{v}_i]_{n_1} - [\mathbf{v}_i]_{n_2}|$ which is low if pixels n_1 and n_2 have similar values for each exemplar in the training set. Greedy clustering using this metric[2] produces the final set $\mathcal{N}$ of query pixels. We call this combination of sorting and clustering "deterministic boosting". Its primary characteristics are fast training and repeatable results. However, its downside is that one needs to be able to deterministically compute the loss function p_n, and similarity metric D which is possible only for simple weak classifiers such as the single-pixel classifiers used here.

In our experiments, with the threshold on D set to the rather tight value of 2, the number of query pixels was reduced from 34788 to 1199, yielding a 30-fold reduction in complexity. Figure 3 illustrates the set of query pixels finally used.

5 Results and conclusions

For testing, we obtained a ten-minute sequence of a typical set of gestures where the user controls a windowed operating system. A gesture is reported to the operating system if it is detected in three successive frames, so the effective frame rate is 5 fps, yielding a test set of 3000 gesture images. The number of false positives reported was 4, corresponding to a 99.87% success rate. The best existing results on comparable (or indeed any) data are those of Birk et al [3] who report a 99.70% success rate (6 failures on 1500 images) for what they term "off-line recognition" on a 25-element gesture set. An in-house implementation of their system requires 208 PCA components to classify our training set, and hence significantly more computation than this paper's proposal.

This paper has described a new approach to hand-gesture recognition which achieves extremely high recognition rates on long image sequences. This is to our knowledge the first

[2] $\mathcal{N} = \{\}$. While P is not empty: (Add head(P) to $\mathcal{N}$. Remove from P all n for which $D(n_1, n_2) < 2$).

demonstration of gesture-based full text entry and mouse operation using computer vision alone.

The performance of the system is achieved by a combination of factors. Careful engineering of the acquisition means that accurate lighting control is replaced by a lightweight passive wristband on the user. This requirement is less intrusive than gloves or finger bands. Secondly, a novel adaptation of the emerging computer vision techniques of exemplar-based recognition and boosting allows a system which is essentially a brute-force template matcher to operate in real time with a large vocabulary.

On the other hand, much work remains. The reason for developing a high-speed training

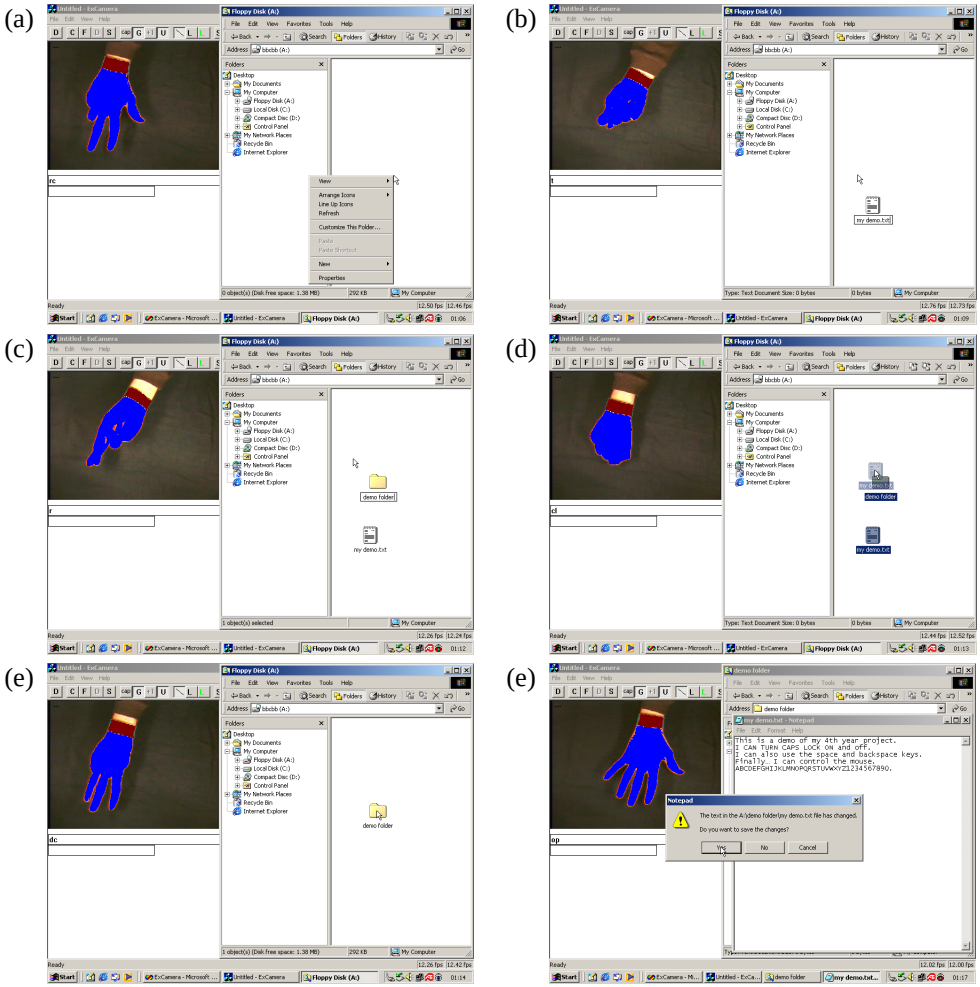

Figure 4: **Demo transcript**. Six frames from a 10-minute demo in which a windowed operating system is fed UI events generated by the gesture recognition system described in this paper (see `http://www.robots.ox.ac.uk/~awf/bmvc02` for the demo). Control is exclusively via hand gestures. Transcript: A right click opens a menu (a), a new text file is created and named `my demo.txt` (b). A new folder is created, named `demo folder` (c), and the text file is dragged into the new folder (d). Navigating to the new folder (e), double-clicking the text file, text is entered, and the file is saved (h). In total, six errors were made, two of which were operator error, and all of which were correctable using the backspace/undo gesture.

algorithm such as deterministic boosting was to allow online retraining: every time a gesture is misrecognized, it would be useful to add the misrecognized example to the training set and retrain. Because both of the clustering algorithms are greedy, it is trivial to add new examples to them. Replacing the greedy clustering algorithms with more sophisticated versions would reduce the number of exemplars, and increase speed further.

Although the system does not require careful lighting, it does depend on its being the same for test and training examples. This is because many gestures are distinguished by fairly subtle shadowing effects. If lighting conditions are expected to vary, the training set should be extended to include examples under all conditions or to switch between training sets captured under the different conditions.

In the future, removal of the wrist band is an obvious candidate enhancement. This will have a number of deleterious effects, particularly if the user is wearing short sleeves, or loose sleeves which move along the arm. In fact, it is fair to say that most existing gesture recognition systems have an implicit "wrist band" assumption—this paper simply makes it explicit.

Acknowledgements This work was supported by the Royal Society and the Department of Engineering Science, University of Oxford.

References

[1] T. Ahmad, C. J. Taylor, A. Lanitis, and T. F. Cootes. Tracking and recognising hand gestures using statistical shape models. *Image and Vision Computing*, 15(5):345–352, 1997.

[2] Anonymous. Hand gesture recognition using computer vision. Technical report, Institution, 2002.

[3] H. Birk, T. Moeslund, and C. Madsen. Real-time recognition of hand alphabet gestures using principal component analysis. In *Proceedings, SCIA*, 1997.

[4] R. Bowden, T. A. Mitchell, and M. Sarhadi. Non-linear statistical models for the 3d reconstruction of human pose and motion from monocular image sequences. *Image and Vision Computing*, 18(9):729–737, 2000.

[5] R. Bowden and M. Sarhadi. Building temporal models for gesture recognition. In *Proc. BMVC.*, volume 1, pages 32–41, 2000.

[6] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Proc. CVPR*, 1997.

[7] R. Cipolla and A. Pentland. *Computer Vision for Human Machine Interaction*. Cambridge University Press, 1998.

[8] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models—their training and application. *CVIU*, 61(1):38–59, 1995.

[9] E. Costello. *American sign language dictionary*. Random House, 1997.

[10] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proceedings*, 1996.

[11] D. Gavrila and V. Philomin. Real-time object detection for "smart" vehicles. In *Proc. ICCV*, pages 87–93, 1999.

[12] T. Heap and D. Hogg. Towards 3D hand tracking using a deformable model. In *Intl. Conf. on Automatic Face and Gesture Recognition*, pages 140–145, 1996.

[13] Jerome Martin and James L. Crowley. An appearance-based approach to gesture recognition. In *Proceedings, ICIAP*, pages 340–347, 1997.

[14] Vladimir Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE PAMI*, 19(7):677–695, 1997.

[15] James Rehg and Takeo Kanade. DigitEyes: Vision-Based Human Hand Tracking. Technical Report CMU-CS-93-220, Carnegie-Mellon Univ, Dec 1993.

[16] S. Romdhani, S. Gong, and A. Psarrou. Multi-view nonlinear active shape model using kernel PCA. In *Proc. BMVC.*, pages 13–16, 1999.

[17] T. Starner, J. Weaver, and A. Pentland. Real-time american sign language recognition using desk- and wearable computer-based video. *IEEE PAMI*, 20(12):1371–1375, 1998.

[18] B. Stenger, P. R. S. Mendonça, and R. Cipolla. Model based 3D tracking of an articulated hand. In *Proc. CVPR*, pages 310–315, 2001.

[19] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *Proc. ICCV*, pages II, 50–57, 2001.

[20] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. CVPR*, pages ??–??, 2001.

[21] C. Vogler and D. N. Metaxas. Parallel hidden markov models for american sign language recognition. In *Proc. ICCV*, pages 116–122, 1999.