CRAVEN, M. P. and CURTIS, K. M., 2004. GesRec3D: A Real-Time Coded Gesture-to-Speech System with Automatic Segmentation and Recognition Thresholding Using Dissimilarity Measures. *In:* CAMURRI, A. and VOLPE, G., eds., Gesture-Based Communication in Human-Computer Interaction: Lecture Notes in Computer Science LNCS 2915/2004. Springer-Berlin / Heidelberg. 231-238. doi: 10.1007/978-3-540-24598-8_21

GesRec3D: A real-time coded gesture-to-speech system with automatic segmentation and recognition thresholding using dissimilarity measures

Michael P. Craven[1] and K. Mervyn Curtis[2]

[1]University of Technology, Jamaica
School of Engineering, Kingston 6, Jamaica, WI
michael.craven@ieee.org
[2]University of the West Indies,
Dept. of Mathematics and Computer Science,
Mona Campus, Kingston, Jamaica, WI
mervyn_curtis@ieee.org

Abstract. In the field of Human-Computer Interaction (HCI), gesture recognition is becoming increasingly important as a mode of communication, in addition to the more common visual, aural and oral modes, and is of particular interest to designers of Augmentative and Alternative Communication (AAC) systems for people with disabilities. A complete microcomputer system is described, *GesRec3D*, which facilitates the data acquisition, segmentation, learning, and recognition of 3-Dimensional arm gestures. The gesture data is acquired from a Polhemus electro-magnetic tracker system, where sensors are placed on the finger, wrist and elbow of one arm. Coded gestures are linked to user-defined text, to be typed or spoken by a text-to-speech engine, which is integrated into the system. A segmentation method and an algorithm for classification are both presented, which includes acceptance/rejection thresholds based on intra-class and inter-class dissimilarity measures. Results of recognition hits, confusion misses and rejection misses are given for two experiments, involving predefined and arbitrary 3D gestures.

1 Background and Motivations

Gestures, comprising complex shapes and movements of the body, are recognised and used effortlessly by humans, enabling a rich communication in combination with visual, aural and oral forms. However, whilst speech and image processing are mature areas in the field of Human-Computer Interaction (HCI), recognition of gestures is still relatively undeveloped. In addition to this, there is great interest in multi-modal HCI techniques that integrate speech, vision and gesture. Gesture recognition is also an important addition to Augmentative and Alternative Communication (AAC) technology, as the needs of many disabled people may be better served by consideration of all potential means of their interaction with computers and communication aids, especially when one or more senses or body functions are impaired. With the increased availability of new input devices for virtual reality, computer games, and computer aided design, there is an even greater need for 3D gesture recognition.

Much of the published literature can be placed in the following categories: extensions of character and speech recognition techniques to 2D and 3D gestures[1,2]; recognition of hand gestures for sign language and other applications, for example[3,4,5]; and mapping of gestures to speech parameters[6,7]. There are also attempts to replace the traditional mouse in software applications, for example gestures drawn with the mouse which replace the function of the mouse buttons[8], or through the use of head gestures[9]. We have previously used 2D projections of arm movements in the *GesRec* system[10].

The need for invariance of one kind or another often determines the type of algorithm e.g. spatial moments for characters, 'dynamic time-warping' for speech. These considerations are especially true in the application of gesture recognition to HCI for disabled users where it may be important to preserve as many degrees of freedom as possible e.g. a person may have limited range of movement, but have a controllable speed for that movement. We make the following observations for gestures (although short of a more rigorous psycholinguistic approach):

- Timing is important. Whereas speech recognition often employs time invariance to account for speaking rate or speaker variability, people can time gestures fairly accurately and use faster or slower forms of the same body movement to good effect.
- Size is important. Whereas character recognition methods often employ size invariance, the size range of natural gestures is quite large, and a large movement can have a different meaning than a small one of a similar shape.
- Translation invariance is still important, although gestures may have different meanings when made in relation to different parts of the body or to other objects.
- Rotational invariance may still be important, although gestures are often made in a fixed relation to another person, and limb movements performed at different angles with respect to the body can have different meanings.

Furthermore, it is often difficult to scale algorithms to 3D, as we have found with our previous implementations of neural networks and dynamic programming, so we are motivated to find computationally simpler matching algorithms such as proposed below. Others have made progress with Hidden Markov Models, for example Hofmann *et al*[11], although training times were still reported to be long, and also with Time-Delay Radial Basic Function neural networks, for which Howell and Buxton used pseudo-inverse weight matrix calculations which avoids stability problems associated incremental learning algorithms like backpropagation[12]. The latter example uses a fixed time window so results differ depending on the speed of the gesture, but the authors state that may be an advantage in distinguishing difference in "intentional force", as we have similarly suggested above.

In this work, however, the main motivation is dealing correctly with unintentional gestures. The proposed method of doing this is to incorporate rejection criteria into both the segmentation and the recognition methods. The latter is achieved by constructing a dissimilarity matrix for the entire gesture set and using this information to calculate acceptance/rejection thresholds automatically. The time taken to compute this matrix is dependent on the complexity of the matching algorithm.

2 Method

2.1 Data acquisition

Real-time gesture information was obtained from a Polhemus 3SPACE FASTRAK six-degrees-of-freedom (6DOF) electro-magnetic tracking system[13], comprising of a transmitter module and 3 sensors placed on the finger, wrist and elbow of one arm (Fig.1). The tracker was interfaced to a Pentium 133MHz PC with a 16-bit SoundBlaster sound card capable of supporting the Creative TextAssist text-to-speech engine.

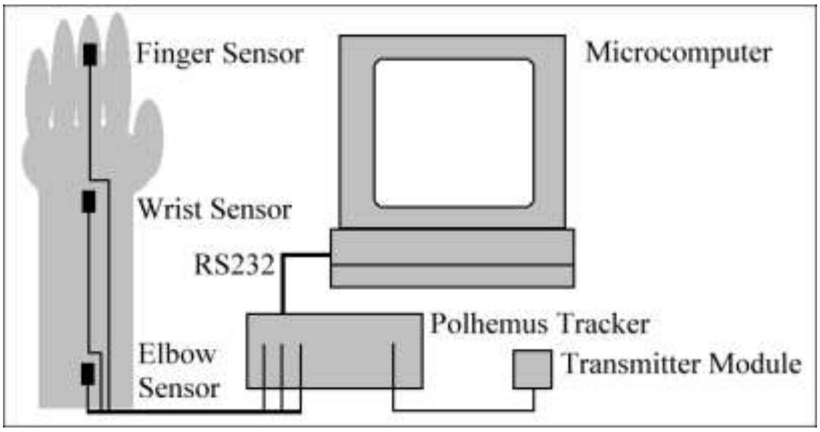


Fig. 1. 3D Gesture acquisition system using a Polhemus 3SPACE FASTRAK electro-magnetic tracker

Software *GesRec3D* was designed to perform the data acquisition from the tracker, by prompting the user for several examples of different gestures in a guided training session. Each different gesture could be freely associated with a text string to be spoken out loud by the text-to-speech engine. The software thus implemented a limited vocabulary gesture-to-speech application. The application was designed to support training and data storage of 5 examples of up to 30 gestures of a maximum 200 samples, at 20 samples a second from each sensor (i.e. 10 seconds of data per gesture), together with storage of all the associated text-to-speech data and training parameters. The system also produced a continuous text format time-stamped log of the raw Polhemus gesture data for superimposing on to video footage of users during training and evaluation sessions.

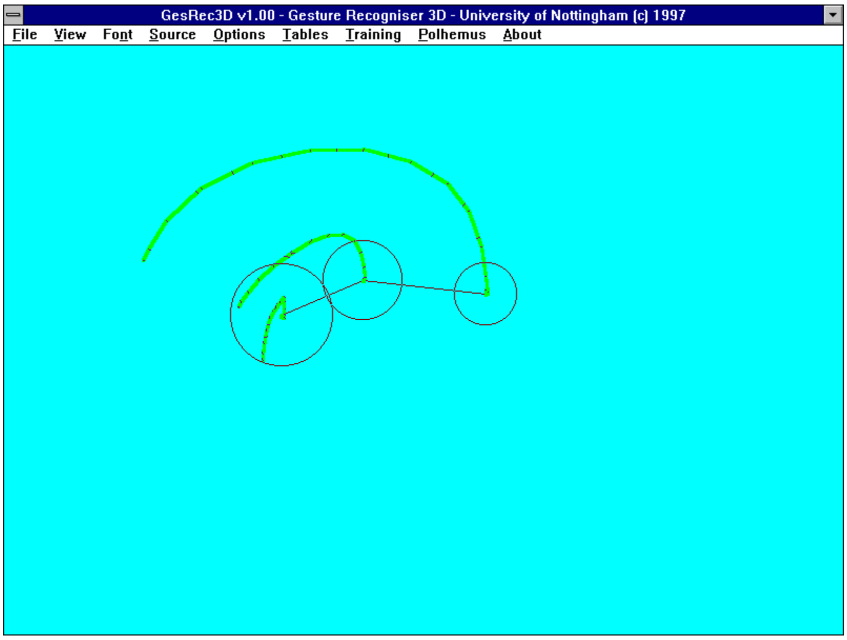

GesRec3D v1.00 - Gesture Recogniser 3D - University of Nottingham (c) 1997
File
View
Font
Source
Options
Tables
Training
Polhemus
About

Fig. 2. Traces from 3 sensors attached to the elbow, wrist, and finger (shown left to right)

Fig. 2 shows the main window of *GesRec3D* indicating a technique we have used to provide visual feedback to the user of their gestures, where movement of each the three sensors in the plane of the computer screen is shown as a 2D trace, and the third axis (distance from the computer screen) is represented as the radius of a circle. The sample spacing gives an indication of speed. When the arm is moved, the movement of the links and changing radii creates an adequate sensation of 3D movement with a minimum of processing.

2.2 Segmentation

A generalised D-dimensional gesture or character can be described after segmentation as a sequence of m co-ordinates,

$$G = g_0 g_1 \ldots g_m \qquad \textbf{(1)}$$

where $g_i = g_i(X_{i1}, .., X_{iD})$.

The 6DOF data available from the FASTRAK consists of both position and angle information. However, an early design decision was taken to use only the (x,y,z) position data from each of the 3 receivers and ignore the angle data, since translation invariance with respect to the transmitter is easily realised, whereas relative orientation angles change with distance. If needed, angles between receivers could be obtained from the position data. Furthermore, when receivers are placed on a person's body by attaching them to arm and finger bands, these may slip round and cause significant errors in measurement of orientation, whereas the effect of this on position is less of a problem.

We make a distinction here between coded and continuous gestures, analogous to isolated characters vs. cursive scripts or isolated words vs. continuous speech. The coded gestures used in this work have explicit start and end states, and so require a segmentation strategy to identify these. We devised such a segmentation strategy using five user adjustable parameters as follows:

1. *Start Gesture Sample Spacing* s_{start} - When the user is at rest the recogniser remains in a start state, where a sample counter is reset to zero. The start of gesture condition is such that the distance between fixed interval samples in any one of the (x,y,z) coordinates must be greater than or equal to s_{start}. This parameter has a small default value so that only a small movement is required to start the gesture, but may be increased for users with continuous involuntary movements e.g. tremor.
2. *End Gesture Sample Spacing* s_{end} - After the start condition is met, the spacing is increased so that a larger movement is required to continue the gesture. If the differences between samples in all of the (x,y,z) coordinates are less than $\mathrm{s}_{\mathrm{end}}$, an end phase is entered, otherwise the gesture is continued.
3. *Minimum samples* m_{min} - This parameter is chosen to ensure that short gestures (with $m<m_{\mathrm{min}}$) are ignored. If the end phase is entered before m_{min} samples have been obtained, the recogniser is immediately reset to its starting state.
4. *End Gesture Time-out* t_{end} - In the end phase, the s_{end} condition must continue to be met for a time $t_{\mathrm{end,}}$ otherwise the timer is reset and the gesture is continued. When the timer times-out, the gesture is ended and recognition can be attempted. The recogniser will have obtained a further $t_{\mathrm{end}}r_{\mathrm{s}}$ samples during the end phase (where r_{s} is the sample rate), which are ignored.
5. *Training Delay* t_{delay} - This parameter is used to allow the user enough time to return to the start position during training. The recogniser is forced to remain in the start state for a time t_{delay}. The parameter is only used in training, so that in normal use the recogniser is always ready for a new gesture.

It should be noted that natural gestures that are characterised by ‘preparation - stroke - retraction’ may have a steady state after the stroke phase is completed which is indistinguishable from an end state. If so, the retraction phase may be ignored, which should not be a problem so long as the stroke phase contains enough information to describe the gesture.

2.3 Dissimilarity measure

One computationally inexpensive dissimilarity measure between *D*-dimensional numeric variables uses the city block metric[14]:

$$d_{ij} = \sum_{k=1}^{D} w_k \mid X_{ik} - X_{jk} \mid \tag{2}$$

where the w_k are used to scale the variables if necessary.

In order to use this to compare equal length sequences of 3D samples, values are accumulated over *m* samples and normalised. As the three dimensions of position have equal weight, only a single scaling factor *W* is needed which is usefully employed in our application to scale the data to integer values and thus speed up computation. Hence, the dissimilarity measure between two gestures of equal length is given by:

$$d = \frac{W}{m} \sum_{i,j=1}^{m} \left(\mid x_i - x_j \mid + \mid y_i - y_j \mid + \mid z_i - z_j \mid \right) \tag{3}$$

In practice, two gestures will be of different lengths, so either the gestures or the measure must be modified to take account of this. If we are employing time invariance, we can interpolate the shortest gesture to equalise the number of samples, and use Equation 3 directly. However, as stated earlier, we would rather use the mismatch in length as a distinguishing factor, so instead we propose the following modification to the measure for two gestures G_a and G_b:

$$d_{ab} = \frac{W}{m_b} \sum_{i,j=1}^{m_a} \left(\mid x_i - x_j \mid + \mid y_i - y_j \mid + \mid z_i - z_j \mid \right) \tag{4}$$

where $m_a > m_b$ and $g_j = (0,0,0)$ for $j > m_b$.

Thus length mismatch is penalised in two ways, firstly by comparing the length mismatched part of the longer gesture with zeros, and secondly by normalising to the smallest length. The measure reduces to Equation 3 when $m_a = m_b$.

2.4 Acceptance threshold

After a training session is completed and the system has acquired enough examples of each gesture class, we can calculate d_{ab} for each pair of gestures. For *C* gesture classes and *n* examples of each class, this yields a square $nC \times nC$ dissimilarity matrix. We now show how to use this matrix to find an acceptance threshold between each pair of classes.

Within each class we find the largest value of d_{ab}, denoted by the worst internal (or intra-class) match d_{int}. For consistently made gestures this value should be small, so it gives us a good measure of repeatability. Between a class and each of the other classes we find a minimum value of d_{ab}, denoted by the best external (or inter-class) match d_{ext}, which should preferably be much larger than d_{int} to prevent confusion between classes.

We are now in a position to define an acceptance threshold d_{th} between any two classes:

$$d_{th} = \frac{K}{2}\left(d_{\text{int}} + d_{ext}\right) \tag{5}$$

For K=1, and $d_{ext} > d_{int}$ this is half way between the worst internal match and best external match and so forms a rejection threshold between non-overlapping classes. However, for very poor matches between classes d_{ext} may be many times greater than d_{int}, making the rejection threshold far greater than necessary. To avoid this situation, an upper bound can be specified for d_{th}.

For $d_{ext} < d_{int}$, the rejection threshold is less than d_{int} but as this is between the least similar examples of a class, a match with another more similar example is still possible.

If K is decreased, the rejection condition is made stricter, and if increased, it can be made less strict. To allow manual adjustment of the threshold, it was decided to add the facility of a global percentage increase or decrease of K to the user interface.

2.5 Recognition

To achieve recognition of an unknown gesture, G, that gesture is matched with every gesture in the training set, repeatedly applying Equation 4 to find the winning class with dissimilarity d_{min}. Considering matches between gestures measured using a single sensor, the recognition process is as follows:

1. Find gesture class corresponding to d_{min}
2. If d_{min}<d_{th} select that class, otherwise reject the gesture
3. Perform action linked to the selected gesture class

For multiple sensors it is necessary to combine the dissimilarity measures from each sensor in some way. We chose to do this by adding the dissimilarity measures obtained for each sensor, and also the corresponding rejection thresholds.

This gives an overall rejection condition, but the problem remains of which class to choose, as the different sensors may not agree. Of course, we could insist that all (or the majority) of sensors do agree, but in practice we found the rejection rate to be too high. Instead we chose the class selected for the finger sensor, as that movement is the greatest, and was considered to be most variable between gestures. The option of whether to use a primary sensor or whether to use the stricter condition of some or all sensors agreeing can be left under the control of the user. Thus the multiple sensor recognition process is:

1. Find gesture class with d_{min} for each sensor
2. (optional: reject gesture if classes are different)
3. Find d_{th} for each class for all sensors
4. Add the d_{min}

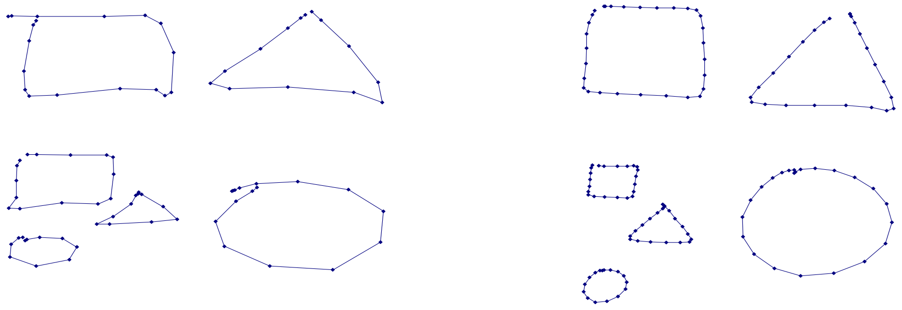
5. Add the d_{th}
6. If $\Sigma d_{min} < \Sigma d_{th}$ select the class corresponding to primary sensor, otherwise reject gesture
7. Perform action linked to the selected gesture class

3 Experiments and Results

3.1 Experiment 1 - 'Shapes'

The first experiment was devised to test the segmentation and thresholding abilities for simple gestures of different sizes and drawing speeds. An able-bodied user was asked to train the system using three different easily remembered shapes - CIRCLE, TRIANGLE, and SQUARE. Each shape had two different sizes, SMALL and LARGE, and for each size two speeds, FAST and SLOW. Thus the training set consisted of a total of 12 gestures, each to be entered 5 times. A table was constructed containing text corresponding to each gesture so that after training the text-to-speech synthesiser would speak the words associated with each gesture e.g. "Small Fast Circle".

It is important from a user's point of view that the effect of changing any parameter is understood and intuitive. Thus, all segmentation parameters were placed together in a dialog box which could accessed from a windows menu item. The start and end sample spacings were presented as integer values in a fixed range of 1 to 20 millimetres, set using a scroll bar control. The default s_{start} was set to 2mm, and the default s_{end} to 6mm. The minimum samples parameter m_{min} was controllable in the range 2 to 30 samples, with a default of 10 so that a gesture of duration less than 0.5 second would be ignored. Time-outs are similarly adjusted, using scroll bars with values at 0.1 second intervals, in the range 0 to 10 seconds. The default end gesture time-out $\mathrm{t}_{\mathrm{end}}$ was set to 0.2 second, and the training delay t_{delay} to 1 second. The global threshold modifier was set to 0, with a range of -100 to 100%. In practice most able-bodied users could use the default values without modification to achieve a satisfactory segmentation. However, in preliminary trials with users with motor disabilities e.g. cerebral palsy, it was generally necessary to increase all the segmentation parameters to some extent, to account for continuous involuntary movements.



(a) FAST

(b) SLOW

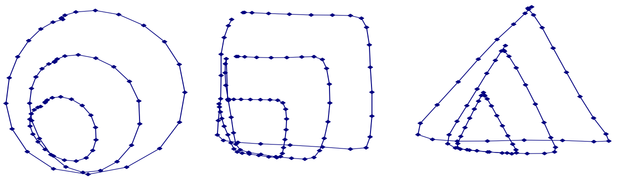


Fig. 3. Examples of gestures used in the shapes experiment, shown as (x-y) 2D projections of the 3D data from the finger sensor

Fig. 4. Examples of LARGE SLOW gestures used in the shapes experiment, shown as superimposed (x-y) 2D projections of the 3D data from all three sensors (finger, wrist, elbow)

Fig. 3 shows examples of the gestures in two groups, (a) FAST and (b) SLOW, captured from the finger sensor at a 20Hz sample rate. All gestures were drawn in a clockwise direction. The slower set of gestures can be identified by the greater number of samples obtained. Fig. 4 shows examples of LARGE SLOW gestures for each shape, simultaneously showing data from all three sensors, in this case mounted on the left arm. The largest traces are for the finger, the medium size traces for the wrist, and the smallest for the elbow.

Training of the system took only 5 minutes for the total of 60 gestures entered. Calculation of the 60×60 dissimilarity matrix took approximately 0.06 seconds per sensor i.e. 0.17 seconds in total for the 3 sensors on the Pentium 133MHz PC. This was much faster than for time invariant matching calculated using dynamic time warping, which took 7.47 seconds, though the aforementioned linear time-invariant version could be computed in 0.6 seconds.

Fig. 5 shows a graphical representation of one row of the dissimilarity matrix computed for the finger sensor, showing the distances (normalised in the range 0 to 1) between one example of the SMALL FAST CIRCLE (SFC) gesture class with all other gestures in the training set. For each class, the best and worst matches to SFC are given. Within class SFC, the best match is 0, resulting from that gesture being matched to itself. The greatest intra-class dissimilarity for SFC is less than the smallest inter-class dissimilarity (in this case for SFT), indicating that SFC is fairly well separated from the other classes, and that examples of that class have good repeatability. It can be observed that the closest gesture classes to SFC are those with similar size and speed, and that after circles, the better matches are obtained for triangles, and then squares. Though the example is somewhat artificial, these results do show the advantage of maintaining both spatial and temporal characteristics of the gestures.

After training, each user was asked to enter each of the training gestures, and also introduce gestures not in the training set. Table 1 shows the results for one user, testing with 50 examples of each gesture classes. It can be seen that recognition hits varied between 82-96% depending on the gesture. There are in general fewer misses from confusion than rejection. In addition 100 arbitrary gestures were made, of which all but one were rejected. More importantly, movements made by the user moving to the start positions of intended gestures (or retracting from the previous gesture) were all rejected. These results do not include any additional small movements rejected during segmentation, as these are not passed to the recogniser.

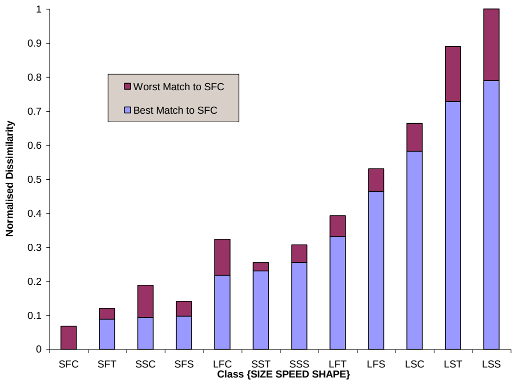


Fig. 5. Intra-class and inter-class dissimilarity measures for the SMALL FAST CIRCLE gesture (data from finger sensor)

Table 1. Recognition results for shapes gestures (50 of each class)

Shape	Hits out of 50	Confusion Misses	Rejection Misses
SMALL FAST CIRCLE (SFC)	45	2	3
LARGE FAST CIRCLE (LFC)	45	1	4
SMALL SLOW CIRCLE (SSC)	44	2	4
LARGE SLOW CIRCLE (LSC)	46	2	2
SMALL FAST SQUARE (SFS)	41	4	5
LARGE FAST SQUARE (LFS)	43	3	4
SMALL SLOW SQUARE (SSS)	45	3	2
LARGE SLOW SQUARE (LSS)	47	1	2
SMALL FAST TRIANGLE (SFT)	44	3	3
LARGE FAST TRIANGLE (LFT)	48	1	1
SMALL SLOW TRIANGLE (SST)	47	1	2
LARGE SLOW TRIANGLE (LST)	48	1	1

3.2 Experiment 2 - 'Greetings'

The second experiment was devised to test the full 3D capability of the system, since in the first experiment the gestures were made in a plane even though all data was processed in 3D. An able-bodied user was asked to make up a set of ten arbitrary gestures using one arm to be used for greeting another person, including two additional gestures to say the greetings or to delete the greetings instead of saying them. After training, the user was asked to make a further 20 examples of each gesture for testing purposes.

Table 2. Recognition results for 3D gestures used in greetings, scored out of 20 (brackets give corresponding results with rejection thresholds decreased by 10%)

Gesture Description	Output Text/Speech	Hits out of 20	Confusion Misses	Rejection Misses
Hand up	"Hello"	18 (17)	1 (0)	1 (3)
Hand up and wave	"Goodbye"	19 (19)	0 (0)	1 (1)
Hand left across body	"How are you"	19 (18)	0 (0)	1 (2)
Hand up & down twice	"Very well"	20 (19)	0 (0)	0 (1)
Hand up & down left diagonal	"Please"	18 (18)	0 (0)	2 (2)
Hand up & down right diagonal	"Thank you"	20 (20)	0 (0)	0 (0)
Point to self	"My name is …"	20 (19)	0 (0)	0 (1)
Point forward	"What is your name"	19 (18)	1 (0)	0 (2)
Point behind	"See you later"	15 (15)	2 (1)	3 (4)
Point from mouth	"Say"	19 (19)	0 (0)	1 (1)
Thumb up	"Okay"	20 (20)	0 (0)	0 (0)
Make a cross shape	"Delete All"	18 (18)	0 (0)	2 (2)

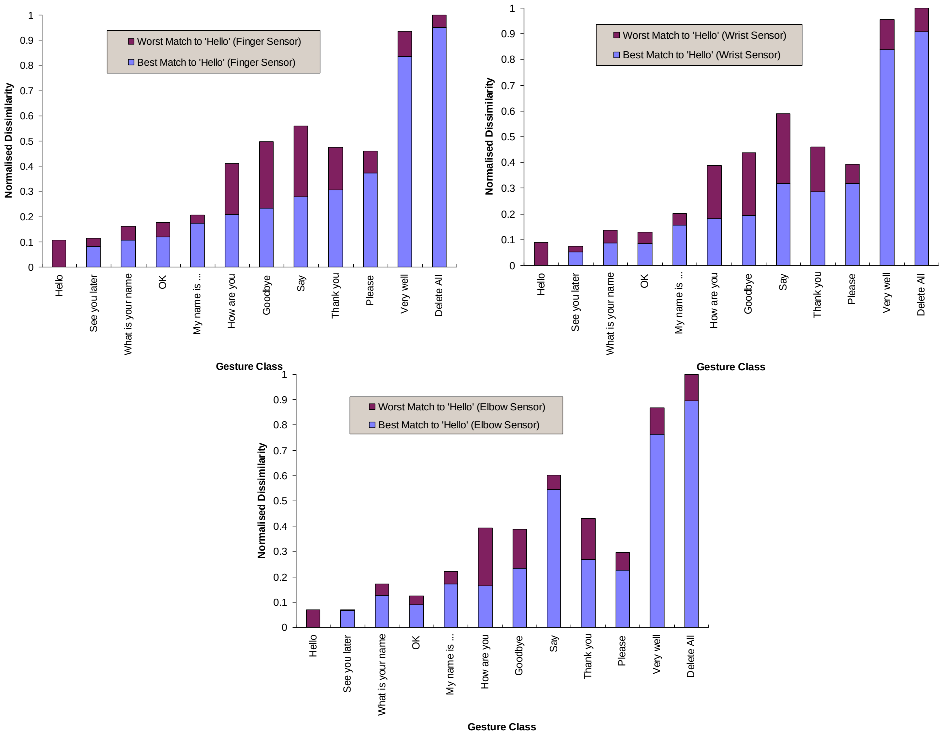


Fig. 6. Intra-class and inter-class dissimilarity measures for a "Hello" gesture using data from finger, wrist and elbow sensors

The set of 12 gestures is shown in Table 2, together with the recognition results. The numbers in brackets are the results obtained when *K* was decreased by 10% using the global threshold modifier. The effect of this is that confusion misses are almost eliminated, whereas rejection misses are increased, and overall recognition rate is slightly reduced.

Fig. 6 shows the values from the row of the dissimilarity matrix corresponding to one of the "Hello" gestures for each of the three sensors. The data is used directly by the software to calculate the rejection threshold for the "Hello" gesture class using the method described earlier i.e. computed from the worst intra-class match for "Hello" and the best inter-class match which in this case is with the "See you later" gesture. The ambiguity of these two gesture classes is reflected in the recognition results, for which several gestures are seen to have been either confused or rejected. The only other gesture class which resulted in a confusion miss in the test set was "What is your name", which is also similar to "Hello". Thus the information obtained from the dissimilarity matrix usefully complements the test data.

4 Discussion and Conclusions

The results from the two experiments show that the *GesRec3D* system is able to acquire and classify gestures that have been learnt by example, utilising both spatial and temporal characteristics of the gestures. The algorithm described gave usable recognition rates (the sample size is too small to say anything rigorous at this point), but more significantly the rejection technique is fast, and allows a user to easily trade-off the recognition rate with rejection rate by the use of the global threshold modifier.

The segmentation method worked well for able-bodied users and also in preliminary trials with disabled users. Some intervention was needed to help the non-technical expert understand the effects of adjusting the segmentation parameters, which suggests the need to automate this as part of the training procedure. It would also be useful to make the parameters less device dependant e.g. independent of sampling rate.

Once the parameters were correctly set, a remaining difficulty was the stamina of the user, especially in training, which in our experience limited the number of gestures trained in one session to between 3 and 5 for the majority of our participants with motor disability. This stresses the need to keep the number of training examples as small as possible, and to allow incremental training. As it stands, the *GesRec3D* system works with only 5 examples of each gesture, and the training data can be saved and reloaded at any point during the training, after which training is continued from the point where the data was last saved. Furthermore, new gestures can be added at any time, and the text or speech corresponding to a gesture can be changed after training if necessary.

Some of the children with motor disability who tried the system were motivated as much by the visual feedback as the speech, and one child even modified a gesture to produce a more pleasing trace pattern on the computer screen. Although visual feedback is not in principle needed for our system to function, the motivational aspect could be looked into further.

Computation of the dissimilarity matrix is fast enough for it to be carried out at program run-time, so it does not need to be stored. We have also implemented modifications of the matching algorithms involving a pre-filtering step using a variable-width averaging filter, in an attempt to remove tremor. In this case it is necessary to recalculate the dissimilarity matrix every time the filter width is changed, which would not be practical if the computation were to take much more than a second. The small computational overhead of the described rejection threshold method provides the means for a user to quickly compare different filter widths.

The graphical information from the dissimilarity matrices indicates which gestures are consistently made, as well as distances between gesture classes. This information could be processed further and used to provide intelligent feedback to the user in the form of suggestions to repeat or change a particular gesture if it is too different from another in the same class, or if it is too much like a gesture in a different class. Such an intelligent system could also suggest parameter adjustments to help improve recognition, or preferably facilitate automatic adjustment.

Recent work has been reported which also utilises dissimilarity measures. Milios and Petrakis used dynamic programming and dissimilarity cost to compare hand shapes for image retrieval, with favourable comparison to fourier descriptors and moments[15]. Long *et al*[16] have used dissimilarity measures for pen gestures using selected features from Rubine[1] and the authors state an interest in creating gesture sets with good memorability and learnability, by relating these to similarity. The latter will be of importance for persons with cognitive impairment in addition to motor disability, and this could also be a useful avenue of research in the field of AAC.

In conclusion, the *GesRec3D* system has provided us with a good test-bed with which to examine improved gesture recognition algorithms. It is intended that future work and user trials will result in a viable user-friendly communication system for people with motor and speech disabilities.

5 Acknowledgements

This work was funded by grant A/P/0543 from the UK medical research charity Action Research for the project: “Improvement of assessment and the use of communication aids through the quantitative analysis of body movements of people with motor disabilities”, whilst the authors were at University of Nottingham, School of Electrical and Electronic Engineering. Results in this paper were previously presented at Jamaica Conference 2000, hosted by the IEEE Jamaica Section. The authors also wish to thank Dr. Clive Thursfield of Access to Communication and Technology (A.C.T.), Regional Rehabilitation Centre, Selly Oak, Birmingham, and the volunteers who took part in the evaluation of *GesRec3D*. A prototype of GesRec3D was demonstrated at A.C.T. on BBC television Tomorrow’s World as ‘A voice for Vicky’ in May 1997. Finally, the authors wish to thank the anonymous referees for their suggestions in aid of revising this paper.

References

1. Rubine, D. Specifying Gestures by Example. *Computer Graphics*, Vol. 25, No. 4, pp. 329-337, July, 1991.
2. Cairns, A. Y. *Towards the Automatic Recognition of Gesture*. PhD Thesis, University of Dundee, November 1993.
3. Harling, P. A. and Edwards, A. D. N. (eds.). *Progress in Gestural Interaction*. Proc. Gesture Workshop ‘96, March 19th 1996, University of York, London: Springer-Verlag, 1997.
4. Pavlovic, V. I., Sharma, R. and Huang, T. S. Visual Interpretation of Hand Gestures for Human Computer Interaction: A Review, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp 677-695, July 1997.
5. Nam, Y. and Wohn, K. Recognition of hand gestures with 3D, non-linear arm movement. *Pattern Recognition Letters*, Vol. 18, No. 1, pp. 105-113, January 1997.
6. Pausch, R. and Williams, R. D. Giving Candy to children: User-tailored input driving an articulator-based speech synthesizer, in Edwards, A. D. N. (ed*.). Extra-*

Ordinary Human-Computer Interaction: interfaces for people with disabilities. Cambridge Series on Human-Computer Interaction 7, Cambridge: Cambridge University Press, 1995, Chapter 8, pp. 169-182.

7. Fels, S. S. and Hinton, G. E. Glove-Talk II - A Neural- Network Interface which maps Gestures to Parallel Formant Speech Synthesizer Controls. *IEEE Trans. Neural Networks*, Vol. 8, No. 5, pp. 977-984, September 1997.
8. Tew, A. I. and Gray, C. J. A real-time gesture recognizer based on dynamic programming. *Journal of Biomedical Engineering*, Vol. 15, pp. 181-187.
9. Keates, S. and Perricos, C. Gesture as a Means of Computer Access. *Communication Matters*, Vol. 10, No. 1, pp. 17-19, May 1996.
10. Craven, M. P., Curtis, K. M., Hayes-Gill, B. R. and Thursfield, C. D. A Hybrid Neural Network/Rule-Based Technique for On-Line Gesture and Hand-Written Character Recognition. *Proc. IEEE Fourth Intl. Conf. on Electronics, Circuits and Systems*, Cairo, Egypt, December 15-18 1997, Vol. 2, pp. 850-853.
11. Hofmann, F. G., Heyer, P. and Hommel, G. Velocity Profile Based Recognition of Dynamic Gestures with Discrete Hidden Markov Models, in Wachsmuth I. And Fröhlich (eds.). *Gesture and Sign Laguage in Computer Human Interaction*, Lecture Notes in Artificial Intelligence 1371, Proc. Intl. Gesture Workshop, Bielefeld, Germany, Sept. 1997, Springer-Verlag, 1998, pp. 81-95.
12. Howell A. J. and Buxton H., Gesture Recognition for Visually Mediated Interaction, in Braffort A. *et al* (eds*.). Gesture-Based Communication in Human-Computer Interaction*, Lecture Notes in Artificial Intelligence 1739, Proc. Intl. Gesture Workshop, Gif-sur-Yvette, France, March 1999, Springer-Verlag, 1999, pp. 141-151.
13. *Polhemus 3SPACE FASTRAK User's Manual*, Revision F, November 1993, Polhemus Incorporated, Colchester, Vermont, USA.
14. Gordon, A. D. *Classification*. Monographs on Applied Probability and Statistics, New York: Chapman and Hall, 1981), Chapter 2, p. 21.
15. Milios, E. and Petrakis, E. G. M. *Shape Retrieval Based on Dynamic Programming*, http://www.cs.dal.ca/~eem/pubs/timproc.pdf, March 23, 2000.
16. Long Jr., A. C., Landay, J. A., Rowe, L. A. and Michiels, J. Visual Similarity of Pen Gestures, *Human Factors in Computing* (CHI 2000), pp360-367.