Finger Detection with Decision Trees

J. Mackie and B. McCane

University of Otago, Dept. of Computer Science.

{jayson, mccane}@cs.otago.ac.nz

Abstract

This paper introduces a fast decision tree based feature classification system for hand gesture recognition and pose estimation. Training of the decision trees is performed using synthetic data and classification is performed on images of real hands. The presence of each finger is individually classified and gesture classification is performed by parts. The attributes used for training and classification are simple ratios between the foreground and background pixels of the hand silhouette. The system does not require the hand to be perfectly aligned to the camera or use any special markers or input gloves on the hand.

Keywords : Decision trees, Gesture recognition, Synthetic training sets, Augmented reality

1 Introduction

This work has focused on producing a gesture recognition and pose estimation system for Virtual and Augmented Reality [1] interfaces. We have been working with the added constraints of unencumbered hands and minimal retraining or calibration between different users. Many previous gesture based systems have the common element of markers on the hand [4, 7], data gloves or coloured gloves worn by the user [6] to allow the gesture and pose to be derived. A system capable of recognising of gesture and pose on unencumbered hands is more accessible to a casual user, only requiring a headset to allow them to interact with the environment.

For many Augmented Reality (AR) systems gesture input is a suitable input mechanism. Even a small set of gestures can allow a reasonable interaction since the user is interacting directly with a complex environment. The most important requirement for a gesture interface is the need to function fast and smoothly. A latency of greater than 1/10 of a second has an increased chance of causing motion sickness and it has been shown that high latencies reduce the users emotional attachment to the environment [8] and the accuracy of motor control tasks is impaired [13].

Gesture recognition systems using markers[4, 7] have the absolute position of some points, and possibly orientation of hand features, they are able to attempt to fit a model to the known information. Methods that attempt recognition on unencumbered hands typically use appearance based attributes and machine learning techniques. Features are extracted from the image and a classification system is trained, most often Decision trees[5, 6], Hidden Markov Models[9] or Neural networks[11]. To date, most of these appearance based systems have used gesture as an input or command issuing system, rather than attempting to interact with and manipulate objects[12] in an Augmented Environment directly with the hand.

We use decision trees because they are able to perform classification quickly. They are a good candidate for a gesture system [5, 6] if an acceptable classification accuracy can be achieved. In an AR system with a limited set of gestures, gestures may be widely separated in the solution space to limit the interference between them at training time.

Gesture may be supplemented by specific features, which can be tracked and positioned in space to provide additional orientation and pose [11, 12] such as finger separation. Even at coarse level this is a richer interface than just gesture, e.g., the amount of gap between the thumb and finger in a pinch motion. This has been implemented using AR markers stuck to the hand in ARToolkit applications [4] and other research [7] allowing a user to pick up and rotate virtual objects that appear to be held by the hand.

This system demonstrates gesture recognition by parts by detecting each of the fingers separately. This finger detection is done in real time, identifying the presence or absence of each of the fingers in a image. It would be equally able to be used to detect the presence of a silhouette feature in another area of the image other than the fingers.

2 Overview

The system has separate training and real-time classification phases, shown in figure 1. Training is performed on processed images from a large image database. Real-time classification is performed on images of hands captured from cameras, processed

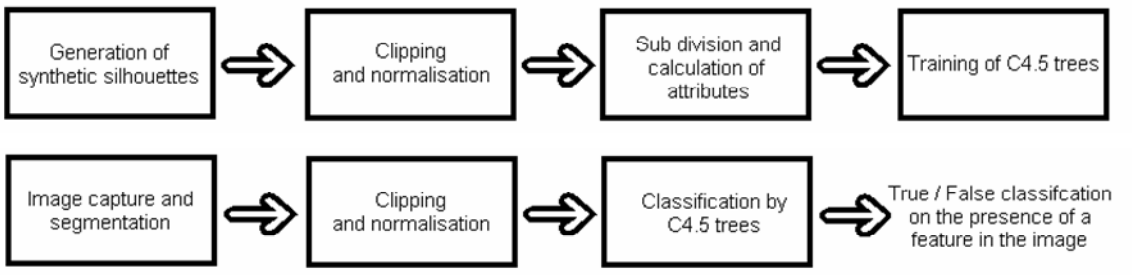

Figure 1: The training and classification processes

and passed to the decision trees generated by the training phase.

A major problem with using machine learning techniques for gesture recognition and pose estimation is collecting enough training data to allow a robust estimation of the model. We solve this in two ways. First, the use of a synthetic hand [2, 6] with 23 degrees of freedom to generate a large data set. Second, we use only the silhouettes of the hands for recognition purposes. This avoids the problem of complex lighting effects which are present in real images but usually missing in synthetic images.

2.1 Image generation

The system used 2280 synthetic hand images. This allowed a large and well defined dataset to be produced for training the trees. Datasets were produced by a ray tracer at a rate of about 750 images per hour. Each finger was moved through its range adduction/abduction of the MCP joint, the side to side motion in the plane of the palm. When the fingers overlapped the positions were altered to make the fingers parallel and the finger tips where never occluded in the training set. Since each gesture was specified algorithmically the ground truth about each pose was easily extracted.

Figure 2: Two images from the synthetic dataset.

The hand was positioned parallel to the camera, filling most of the frame, in the ideal classification position. This ensured that the maximum number of pixels were available before normalisation of the training data.

2.2 Image segmentation

Image segmentation of the hand is not addressed in this project. The training images are produced by the ray tracer as a silhouette. The real hand test images have been photographed against a black background and thresholded to 1 bit images. Any clean up of the test images required by noise or foreign objects at the edge of the test photograph have been done manually.

The live system has a black covered workspace and a pair of black sporting wrist bands are used to produce an easily segmented disembodied hand for further processing. The hand itself remains unencumbered. Any arm observed which is not part of the hand blob is considered on the other side of the wrist band and ignored. Thresholding is performed to convert this to a 1 bit image for input to the classification system.

2.3 Image normalisation

After being generated by the ray tracer or captured and thresholded from the camera the images were clipped to the extents of the hand silhouette and then normalised to a unit square image. The normalisation of an input image has been used in other gesture and pose estimation systems[5, 9] to convert the input image into a template approximating the shape of the hand. If the original image is small the normalised image cannot contain any more information than the original, as shown in figure 3. This system used a 256x256 pixel normalised image to guarantee all possible information could be extract from the 320x240 input images.

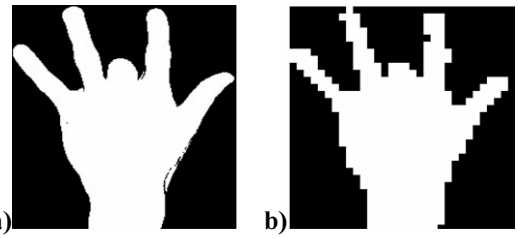

Figure 3: a) An optimal image. b) A small image scaled up from a small source to produce a unit image.

This process allows the system to handle hands presented at an angle to the image plane of the camera. Once normalised, images present an image close enough to the trained data to allow classification.

Figure 4 shows the normalised images of the hand taken parallel to the image plane and at a rotation of 30 degrees off parallel through both the vertical and horizontal axis of the plane of the hand. The shape variation is no larger than the difference between

hands of different users in the finger region but larger in the area of the back of the hand and base of the thumb.

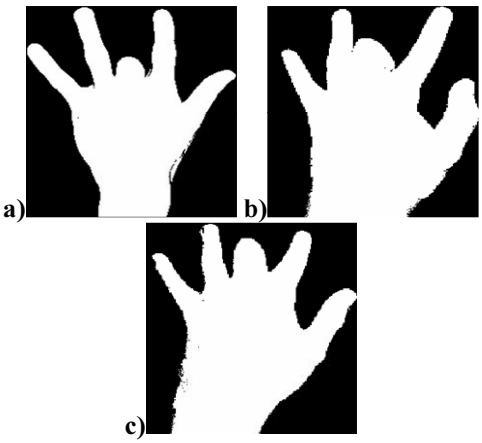

Figure 4: a) A hand parallel to the image plane. b) A hand with vertical axis rotation. c) A hand with horizontal axis rotations, all images normalised.

2.4 Image subdivision

Each image is divided into several sub-images in addition to using the full image. The sub-images are created by dividing the full image into regions of half and a third of the height and width of the normalised image. Additional sub-images overlapping the primary sub-images are also generated. Rectangular sub-images, 2/3 and 3/5 of the height of the image, at layers 1 and 2 respectively are also generated.

At each layer n, n>=0, the number of square images generated is $(2n+1)^2$ and the number of rectangular images is 2n(2n+1). We used 2 layers of sub images, therefore from equation (1) where l=2, we have 56 images, 35 square and 21 rectangular.

$$\text{Total images} = \sum_{n=0}^{l} (2n+1)^2 + 2n(2n+1) \quad (1)$$

Figure 5 shows the square (S1-S9) and top half of the rectangular (R1-3) sub-images for layer 1. Figure 6 shows the non overlapping square sub-images of layer 2.

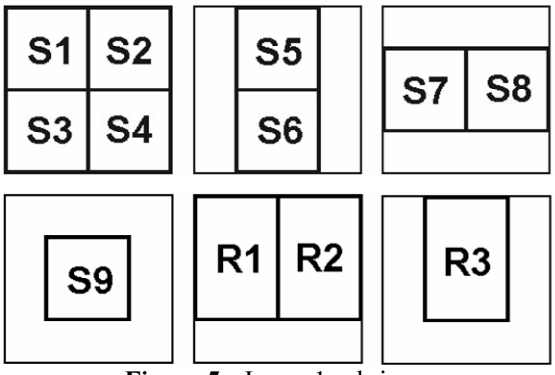

Figure 5: Layer 1 sub-images.

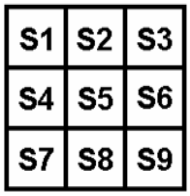

Figure 6: Layer 2 sub-images.

2.5 Image Attributes

For both the synthetic training images and the real images a fast numerical analysis is performed on each of the sub divided zone of the image. The images produced for layers 0-2 generate a 224 element dataset by generating 4 values for each area. The four values calculated are,

- Centre of mass of the foreground area in the X axis, scaled 0.0 to 1.0
- Centre of mass of the foreground area in the Y axis, scaled 0.0 to 1.0
- Ratio of foreground pixel to the total number of pixels in the area, range 0.0 to 1.0
- Ratio of edge pixels in the foreground of the sub-image to the length of the edges of the area, no scaling.

The values for centre of mass give a measure of the position of the hand in the image. The foreground pixel ratio is the primary measure of the presence of a feature in an area of the image. The edge ratio is able to distinguish between two fingers in contact and slightly separated. The length of edges will increase while centre of mass and foreground pixel ratio will remain relatively constant.

The overlapping image subdivision lowers the instability the decision tree would have around a split point. A feature moving primarily from one area to another is represented in the area overlapping both areas, allowing the tree to represent the transition on two branches. e.g. In figure 4 a feature moving from being present primarily in square sub-image S1 to S2 will be present in sub-image S5 throughout the transition. This may contribute to larger trees, but lower errors should result from fewer sudden switches in decision state of the tree.

2.6 Classification using C4.5 Decision trees

The decision tree implementation used was the J48 class of C4.5 decision trees provided in the Weka machine learning environment [14, 15].

One classifier was trained for each finger and the thumb. The classifier returns true or false to indicate the presence or absence of a finger in the image.

Each classifier is constructed by bagging seven C4.5 trees. The seven trees were generated from the

synthetic data set. Sequential, non random sections of the dataset were used to generate the trees. The parts of the dataset used were, the full data set (one tree), each half of the data (two trees), and each quarter of the data set (four trees). This split was chosen because of the distribution of the dataset. The position of the fingers ranges from maximum adduction at the start, rotation in the plane of the palm away from the thumb, to full abduction at the end of the dataset, rotation towards to thumb. This distribution results in the mass of the fingers tending from left to right in the training images, the sequential division of the training set may be able to use local attribute distribution that are not present globally.

The trees in each classifier were initially constructed using the default Weka parameters. Classifier performance varied between 68% and 84%. It was observed that changes to the tree parameters had significant effect on performance. For each of the five classifiers the effect on performance from changing tree parameters was sampled. The optimal parameters for the trees of each classifier were selected. A variation of 40% was observed between the best and worst parameter selection for each classifier. The parameters changed were, the minimum number of items per leaf, confidence factor for the J48 tree pruning algorithm, number of folds used for cross-validation, and the number of votes required for a true to be returned. Bagging is normally implemented requiring a simple majority of votes required, our parameter search showed this did not always generate the optimal performance on our test set. Using the optimal parameters a finger classifier is returns *true* for an input image *I*, when the sum of *true* votes from the seven decision trees equals or exceeds the vote threshold for that classifier, δ_f (2).

$$C_f(I) \equiv \sum_{n=0}^{6} T_{n(I)} \geq \delta_f \qquad (2)$$

The optimal parameters for each of the seven trees in a classifier were not individually searched. The performance variation observed by varying tree parameters for all seven trees uniformly suggests that it is unlikely to be optimal for all seven trees. A search of each of the seven solution spaces for the five classifiers has not been performed. This could be expected to improve the classification performance of the system.

3 Results

The test dataset contains 190 images in 8 datasets from 6 different people. Where multiple datasets have been used from the same individual they have been sourced at different times and accordingly contain some variability in pose and orientation of the camera.

	Correct (%)	False Positive(%)	False Negative(%)
Thumb	98.2	0.00	1.8
First	87.5	12.5	0.0
Index	94.6	5.4	0.0
Ring	87.5	8.9	3.6
Pinky	92.7	5.5	1.8

Table 1. The results of classification on the test images.

These results can be achieved at speeds that allow the technique to be used for real time gesture input. Training is able to be performed quickly and can be absorbed as part of the startup time. Once training is complete the evaluation of 5 classifiers at each frame runs faster than the frame rate (30fps) of the camera system without significant CPU loading (~20%). The system used was a Pentium 4 2.8Ghz.

The use of decision trees allows us to inspect the cases that produce incorrect results. This may enable us to use other sub-image areas or validation system to identify problem gesture. An example is the error from the classification of the first finger, adjacent to the thumb, which could be observed to be using the presence of silhouette in the area at the top of the image, where the finger tip would be when the finger was extended. When a user has a several finger bent the normalised image, as shown in figure 7, still has the 'tip' of the finger in the same location. This indicates a deficiency in the training data.

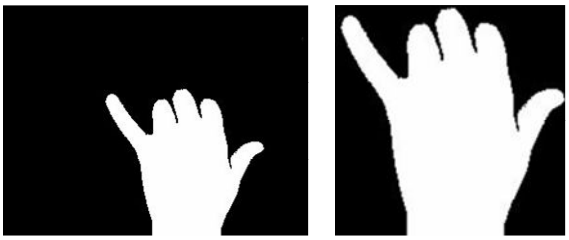

Figure 7: This is incorrectly classified as a positive first finger because the normalised image(right) still has the end of the first finger in a reasonable position for a true classification.

4 Future work

The next stage of development is incorporating the current method into a simple AR environment. A system using a simple gesture set of pinching, pulling and pushing objects with unencumbered hands will be implemented to provide a test bed allowing further developments to be tested in their target system and investigate the positioning of additional cameras to observe the workspace and the accuracy of gesture pose and position estimation.

Other training datasets will also be looked investigated, larger datasets with adjacent pairs of absent fingers and the 32 images created from binary counting with the fingers will be explored.

The limit of hand rotation at which image normalisation is no longer able to produce a reasonable hand image causing performance to be degraded is to be investigated. This measure will allow the user to be aware of the range of motion through which the system should be able track their gestures. Requiring a degree of user goodwill with respect to not intentionally confusing the system will be a constraint of the initial system.

5 Conclusion

We have demonstrated a decision tree classification system trained on synthetic data able to classify images of real hands. The detection of fingers with decision trees provides a fast classification system suitable for hand gesture and pose input for real time systems. The system is able to detect the presence or absence of each of the fingers in a hand image. This may be used as either primary information, combined as identifiers for gestures, or to provide hints to a model based hand pose estimation system if high accuracy joint angles are needed.

Training the trees with synthetic data sets, performing normalisation of training images and evaluating overlapping regions of the normalised images gives smooth classification performance throughout the angle of motion for each finger and moderate rotations of the hand relative to the camera plane.

Training time is a few seconds for each tree, making it acceptable to train at start up. This allows different datasets to be loaded at start-up. Once training is complete the evaluation of 5 classifiers at each frame runs at 30fps without significant CPU loading.

6 References

[1] ARToolkit. http://www.hitl.washington.edu/artoolkit/ (August 2004)

[2] A. Athitsos, S. Sclaroff: An Appearance-Based Framework for 3D Hand Shape Classification and Camera Viewpoint Estimation, In *Proc. Fifth IEEE International Conf. on Automatic Face and Gesture Recognition,* 2002, Washington D.C.

[3] L. Breiman: Bagging predictors, *Machine Learning*, 24:123-140, 1996.

[4] V. Buchmann, S. Violich, M. Billinghurst, A. Cockburn: FingARtips: gesture based direct manipulation in Augmented Reality. In Proc. of the 2nd international conf. on Computer graphics and interactive techniques in Australasia and South East Asia (Graphite 2004). 15-18th June, Singapore, 2004, ACM Press, New York, New York, pp. 212-221.

[5] S. Gutta, J. Huang, I. Imam, H. Wechsler: Face and Hand Recognition Using Hybrid Classifiers, In *Proc. 2nd International Conf. on Automated Face and Hand Gesture Recognition (ICAFGR'96)*, 1996.

[6] Y. Iwai, K. Watanabe, Y. Yagi, M. Yachida: Gesture Recognition by Using Colored Gloves, *IEEE International Conference on Systems, Man and Cybernetics (SMC'96)*, Vol. 1, pp. 76-81, Beijing, China, Aug. 1996.

[7] Y. Kojima, Y. Yasumuro, H. Sasaki, I. Kanaya, O. Oshiro, T. Kuroda, Y. Manabe and K. Chihara: Hand Manipulation of Virtual Object in Wearable Augmented Reality, In *Proc. 7th International Conf. on Virtual Systems and Multimedia (VSMM'01)*, pp 463-470, October 2001

[8] M. Meehan, S. Razzaque, M. Whitton and F. Brooks Jr.: Effect of Latency on Presence in Stressful Virtual Environments, In *Proc.2003 IEEE Virtual Reality Conf. (IEEE VR2003)*, pp 133-140, March 2003

[9] K. Oka, Y. Sato, and H. Koike: Real-time Tracking of Multiple Fingertips and Gesture Recognition for Augmented Desk Interface Systems, In *Proc. 2002 IEEE International Conf. on Automatic Face and Gesture Recognition (FG 2002)*, pp. 429-434, May 2002.

[10] J. R. Quinlan: Bagging, Boosting and C4.5, In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pp. 725-730. Menlo Park, CA: AAAI Press, 1996.

[11] Y. Sato, M. Saito, and H. Koike: Real-time input of 3D pose and gestures of a user's hand and its applications for HCI, In *Proc. 2001 IEEE Virtual Reality Conf. (IEEE VR2001)*, pp. 79-86, March 2001

[12] J. Segen and S. Kumar: Shadow Gestures: 3D Hand Pose Estimation using a Single Camera, In *Proc. IEEE International Conf. on Computer Vision and Pattern Recognition (CVPR),* Fort Collins, June 1999.

[13] B. Watson, N. Walker, P. Woytiuk and W. Ribarsky: Maintaining Usability During 3D Placement despite Delay, In *Proc.2003 IEEE Virtual Reality Conf. (IEEE VR2003)*, pp 133-140, March 2003

[14] Weka Machine learning project homepage, http://www.cs.waikato.ac.nz/~ml/ (August 2004)

[15] I. Witten, E. Frank: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, October 1999.