2006

# Pattern matching techniques applied to human computer interfaces for virtual environments applications

Rami H. Khouri
*Lehigh University*

Khouri, Rami H.

Pattern Matching Techniques applied to Human Computer Interfaces for Virtual...

May 2006

# Pattern Matching techniques applied to Human Computer Interfaces for Virtual Environments applications

by

Rami H. Khouri

A Thesis

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in
Computer Science

Lehigh University
May 2006

This thesis is accepted and approved in partial fulfillment of the requirements for the Master of Science.

April 27 2006
Date

[illegible]
Dr. Henry Baird
Thesis Advisor

[illegible]
Dr. Hank Korth
Chairperson of Department

# Preface

# Abstract

An experiment in using pattern recognition techniques in Virtual Environments (VE, also known as Virtual Reality) is described. We wish to increase the range of commands that a user can issue to a VE system in order to accommodate the lack of a keyboard. Specifically we have tried to provide commands corresponding to all 26 English language characters. Ideally a solution presented for this purpose should be fast, accurate, require little to no training, and provide minimal stress to the user. Our proposed solution is to use pattern matching techniques to identify specific gestures made by the user then execute the function that corresponds to that gesture. This solution is a continuation of the previously published work "Connecting the Dots: Moving towards Text Input in Immersive Environments" [1], which describes a Text Input widget based on a pen and tablet was designed and implemented at Lehigh University.

# Acknowledgement

Dr. Henry Baird: After hearing a very rough proposal from the author Dr. Baird (an expert in the field of pattern recognition) volunteered to act as thesis advisor for this work, and in the process taught the author the basics of pattern recognition and lent his tremendous expertise in forming the focus and direction of the research at every step of development. I thank you for your advice and your guidance, but mostly I thank you for your encouragement.

Dr Scott Frees: Friend and colleague, Dr. Frees was pursuing his Ph.D. from Lehigh University and researching in the GIVE (Graphics Interaction and Virtual Environments) Lab concurrent to the writing of this thesis. Dr. Frees was a co-developer of the "Connect the Dots" interface [1] which served as a spring board for this thesis, and contributed from his knowledge and experience in the field of Virtual environments to this thesis. I thank you for your help and your contributions, but mostly I thank you for your example.

Dr. Drew Kessler: Teacher, advisor and mentor. Dr. Kessler was one of the authors of the Simple Virtual Environments (SVE) software which is the basis of the GIVE Lab (also founded by Dr Kessler). After attending some of Professor Kessler's lectures. I asked for permission to join his lab in the capacity of independent studies. Dr. Kessler's decision to allow me to join his lab was the first of a series of steps that steered my education at Lehigh in a research oriented direction culminating in the writing of this thesis. I thank you for giving me direction and for the things I learned from you, but mostly I thank you for the chance you gave me to prove myself.

Rami Hani Khouri

# Outline

# Chapter 1

# Introduction

## 1.1 Motivation

Since the inception of the field of computing has progressed on a very steady state in terms of processing capabilities and along with the increase in power came software with an expanding range of functionality for the user. Through all this what has remained relatively constant is the interface between User and Computer. The Keyboard, Mouse, Monitor, and speakers have become standard equipment on any PC, and almost all interactive pc software is dependant on their input. As the amount of information a program can process increases, so does the required amount of input from a user. Given the rate at which processing capability has been increasing, and the relative static nature of the standard interfaces, the Keyboard (a 125 year old widget originally designed to slow down typists that were fast enough to jam a typewriter) and mouse have often been accused of being a bottleneck between a user, who can think of detailed commands quickly and the computer with can process them very quickly. This has motivated the search for alternate Human Computer interfaces in general.

Specifically, a changing model of computing has given presented a more urgent need for developing new human computer interfaces. The popularity of Cell Phones, their increased sophistication and SMS text messaging capability along with the development of the market for Personal Digital Assistants (PDA's) have motivated the search for alternative text input. The purpose of the experiment presented in this paper is to investigate text and command input solutions specifically for the domain of Virtual Environments, attempting to overcome one of the hurdles that is keeping VR applications away from mainstream use.

## 1.2 Suggested Solution

We will investigate the feasibility of using the stylus as a pen to "draw" gestures that correspond to certain functions (passing a decision to an underlying system which can implement the function). The gestures will not be restricted before hand to any certain size or speed (larger, clearer motions or smaller, faster ones are both acceptable). Also we will not restrict gestures to any certain dimensionality: that is the gestures do not have to lie in a two dimensional plane; they can be full three dimensional gestures. The main focus of this paper is to test this approach on the 26 English language characters (as it was with [1]) and report on the findings.

## 1.3 Approach

The approach taken in this thesis is start with a base experiment consisting of only a few classes (types of gestures), then based on success or failure expand the scope of the data presented in an iterative fashion, until a set of classes including gesture for each letter of the alphabet is present. At first techniques with low CPU cost overhead are applied, then when necessary more complex decision techniques are added.

Due to time restrictions, and lack of permission to test with human subjects, the primary user this system was tested with is the author of this paper. Some very limited tests were taken with secondary subjects, but the results of them cannot be shared in this paper.

# Chapter 2

# Virtual Environment Standard Equipment.

The standard Virtual Environments interface consists of three main parts. Head Mounted display unit, Stylus and tracker.

## 2.1 Tracker

A device mounted in a fixed position, ideally to an immovable object, floor or ceiling. The tracker samples the position of tracking units attached to certain objects (such as the HMD and Stylus described below) and is used to translate their position into the virtual world. All coordinates (positions of tracked objects and their orientation) are projected to the user relative to the position of the tracker (moving the tracker would result in moving the entire virtual world from the user's perspective). The tracker used in this experiment had a tracking rate of 120 samples per second. These 120 samples will be distributed, round robin to each of the devices being tracked. If there are two objects to be tracked, then the position of each will be measured 60 times per second. If there are four then 40 times per second, three objects will results in 30 measurements of position for each object…etc. In other words the higher the number of objects present, the lower the rate of tracking or the "resolution" of the movement. The Tracker used in this experiment had a reliable tracking radius of four feet, increasing in accuracy when the objects were closer to the tracking device.

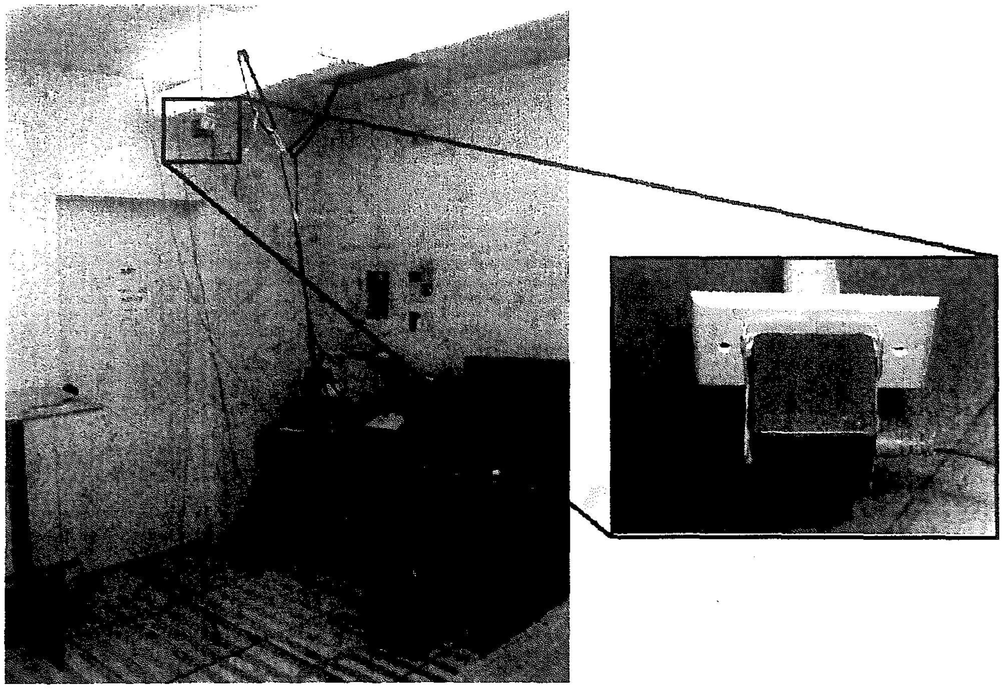


Figure 2.1

## 2.2 Head Mounted Display (HMD)

The head mounted display is analogous to a Monitor in the traditional desktop. The HMD consists of two small monitors connected to a strap that a user can attach around his head such that the monitors align with the users eyes, one monitor over each eye. The HMD also has a tracking unit, an attachment that is used to keep track of the position of the head of the user. based on the position of the users head and image is rendered to correspond to what the user would see if he/she were looking into the virtual world. If the Virtual environment has a stereo configuration then the images presented in each eye differ slightly to represent the image that would be visible by that specific eye (left eye as opposed to the right eye). This is done to give the user a sense of depth perception in the virtual world. The HMD can also be equipped with headphones or augmented with speakers to provide audio sensory feedback to the user from the Virtual Environment. The HMD is analogous to the Monitor and speakers.

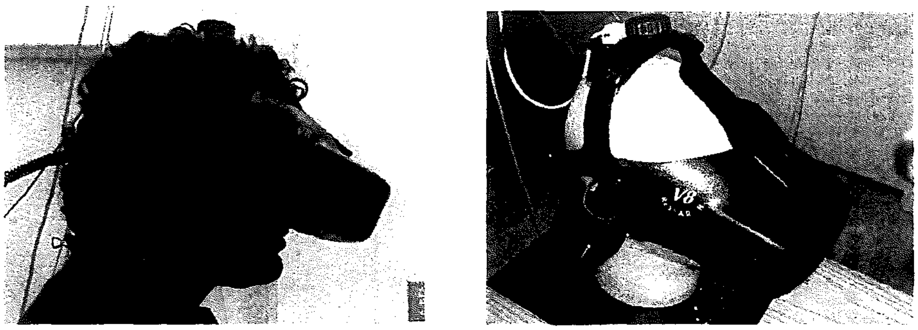


Figure 2.2

## 2.3 Stylus

a pen shaped widget with one button (standard configuration) and a tracker. This widget is meant to be held in the user's hand. There is normally a Virtual object that corresponds with position of the stylus in the real world. This Virtual object is a visual indicator of the position of the stylus which aids the use of the stylus as a tool of interaction with the virtual world. The user can point out objects in the virtual world and a combination of placement of the stylus and interaction with the button normally implies a command by the user. For example, placing the stylus within the space of a Virtual Object and depressing the button may imply "holding the object" (that the object should move along with the motion of the stylus) while releasing the button can imply that the user has "let go of the object" (that the object should now remain in place). In most VE applications the stylus is the sole source of command input to the system from the user, and as such must be overloaded with more than one function. The stylus is analogous to the mouse.

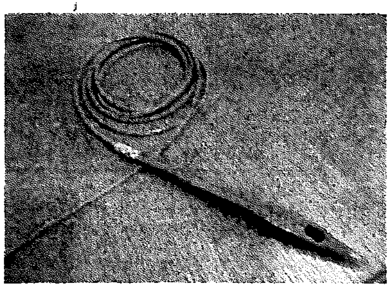


Figure 2.3

## 2.4 Rendering Machine

The central piece to any VR setup is the computer where the position information from the tracker is interpreted, and the images representing the virtual world are rendered and sent to the HMD. The actions of the objects of the virtual world (scripted or otherwise) are processed by this machine, as is the interaction between the user and the virtual world. The software used for this experiment was the Simple Virtual Environments API, available at Lehigh University.

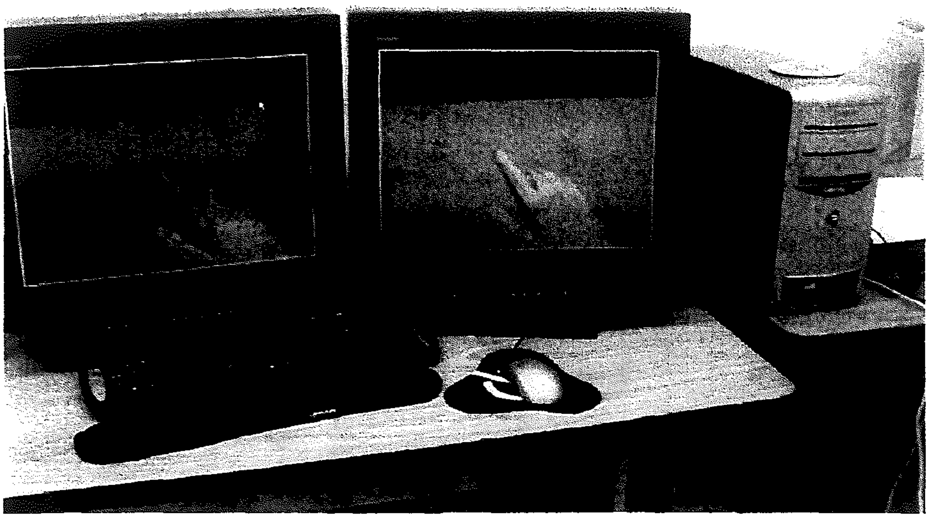


Figure 2.4

This experiment used the Simple Virtual Environments (SVE) library to render the worlds, and retrieve coordinates from the tracker.

## 2.5 Putting it all Together

All these components work together to give the user the ability to navigate and manipulate the virtual world. . The tracker provides the rendering machine with the position of the head and stylus. The rendering machine broadcasts an image to each of the monitors in the HMD. allowing the user to look at the virtual world in three dimensions. The image broadcasted corresponds to the view of the user according to his position as recorded by the tracker. The user can interact with the world by changing the position of the HMD (moving his head. and consequently his viewing area) and the position of the stylus (moving his hand). The user may also have access to some buttons (there is one on the stylus. and the user may also hold a device with buttons on his non-dominant hand). The combination of all three devices allows the user to be "immersed" in the virtual world and interact with the objects placed within.

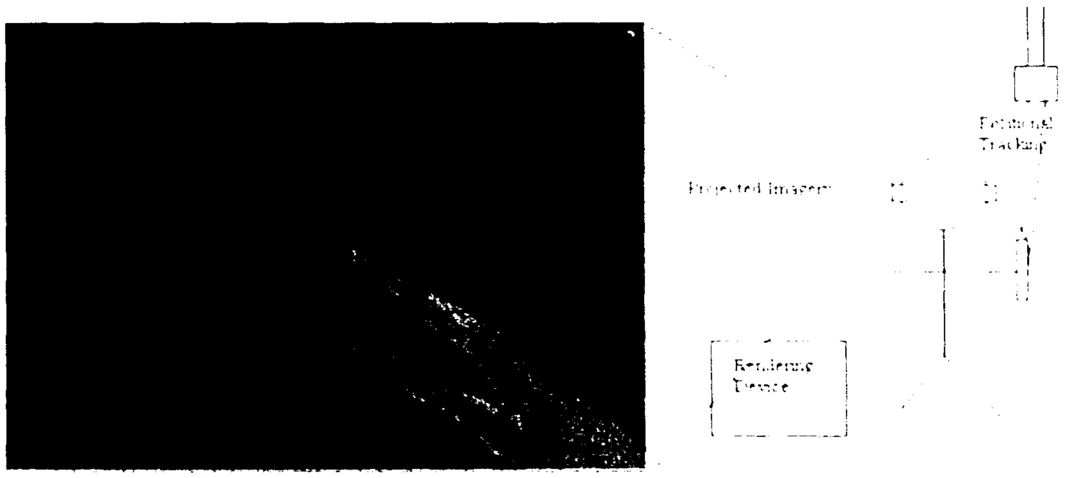


Figure 2.5

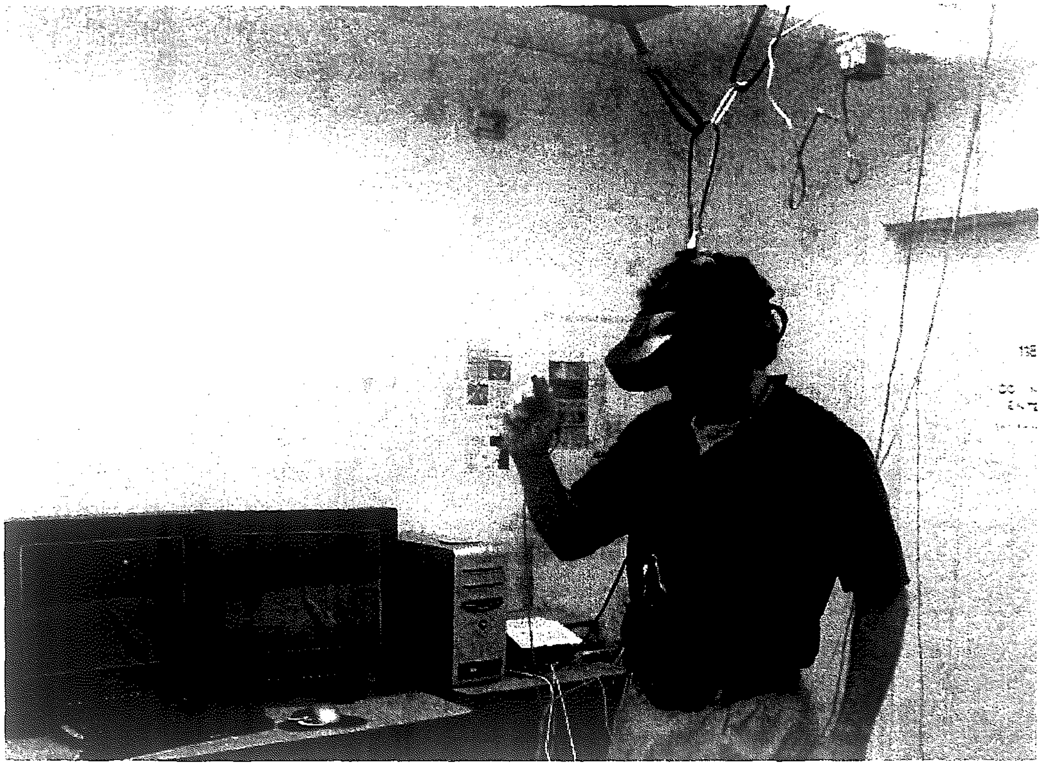


Figure 2.5

# Chapter 3

# Collecting the Data

Before the classification could begin, a supervised learning process has to give the classifier enough samples to build a case to use to discriminate between gestures. In order to describe a gesture a certain notation must be defined, then this notation is used to record a set of truth labeled samples (examples where the intended gesture was marked).

As stated in the introduction, the data was collected using one primary subject, who models the expert VR user. A smaller set of data was collected from secondary subjects, but due to constraints in time and permission for testing with human subjects, this data was not used extensively in testing.

## 3.1 Modes of Interaction

One question briefly investigated here is the manual interaction between the user and the text input widget will take place. Several models were suggested as to how the classifier would receive its input from the user of the VE application

### Tablet Based

Motivated by the "Connect the Dots" text input interface developed at Lehigh University, this interface designated certain areas as "gesture input" areas. These areas are marked by semi transparent 3d objects (flat rectangles) in the virtual world. The objects can be attached to a fixed position, perhaps on top of a physical surface to provide haptic feedback as was done with the CTD, or Attached to the users hand using a tracked board as was done in certain experiments on proprioception [14]. Both of these techniques have been shown to greatly improve the accuracy in which the stylus is manipulated, both for fine manipulation of the stylus position and repeating movements more consistently. The widget would begin recording the gesture when the stylus enters the area, and stops as when the stylus is detected outside the area, and the gesture is sent to the classifier for classification.

### Button based

A very straightforward way of identifying intended gestures to the classifier would be to use the button on the stylus. Depressing the button would indicate the beginning of a gesture. The user would hold the button down while he motioned through a gesture, then release the button when the motion is complete. This was the interface used for the majority of data collection for this experiment. The major down fall of this method is that most VE applications already overload the stylus button, which has a standard function of attaching a virtual object to the stylus (in other words, allowing the user to hold an object and move it if he were holding it with his hand), it would be ideal not to overload this button with more function.

## Constant Listener

In this model is more technically demanding of the classifiers accuracy. It requires that the classifier receive a constant stream of positional data from the VR application. The Classifier will receives motions that represent gestures as well as motions that are not meant to be interpreted as gestures (such as using the stylus to move an object, moving the stylus while the user navigates the virtual world, pointing in a direction for another user to see in a shared application….etc). It would then be up to the classifier to recognize which gestures are to be interpreted as commands and which to ignore. To this end the next section includes the addition of a "none of the above" class representing the motions that are to be ignored.

## Combination

Finally, we can combine any of the above modes of interaction, depending on state of the world. Because some forms of interaction scale better than others, it is suggested that there could be a hierarchy of modes of interaction. A constant listener model can be the default form of interaction, and be available on top of the any functionality that the application provides by default without having to overload more functionality on the stylus button (which is probably already mapped to other functions, like "holding" virtual objects, "pressing" virtual widgets...etc). Since the constant listener mode is less accurate than other forms of interaction (due to segmentation issues & having to differentiate gestures from noise) the number of classes can be limited to a small set of "Gateway" classes. ). A gesture indicating that the user wants to input a series of gesture commands could activate a more accurate form of input than the constant listener, which can handle more classes. For example, a gesture can indicate that the user is ready to input some text, which would bring up the tablet interface. The classes selected as gateway features would be the ones with a small error rate, and can be checked redundantly with multiple classifiers or feature sets (described in the following sections). Because the number of classes would be low, redundant checking would not be as costly as with cases with high number of classes. Special cases could also involve extra input from the user, for example a major command such as "delete" could require the user to input two sequential "delete" gestures, or a delete gesture followed by a gesture to confirm the action (as is suggested in NORMAN). The chances of false positives should be minimized by tuning the classifier (more discussion on this in chapter X). if this is achieved then the chances of two false positives in a short period of time should be very small.

## 3.2 Data Format

In describing a gesture, relevant data was extracted using a frame call-back function activated once every frame. The data that was polled is as follows:

- Position of the stylus in 3 space (x,y,z values), relative to an origin of the virtual world
- Orientation of the Stylus recorded in pitch, yaw and rotation (x,y,z values)
- Position of the HMD in 3 space (x,y,z values), relative to an origin of the virtual world
- Orientation of the HMD recorded in pitch, yaw and rotation (x,y,z values)
- Relative timestamp at which the frame was rendered (integer of CPU cycles)

The information collected in an individual frame's sampling was grouped together as a "moment" of the motion, the coordinates at a specific period of time. Grouped together, a series of consecutive moments makes up a "motion". The data was stored in an external file for evaluation separately at a later date. Each moment took up one line of text, consecutive moments followed each other separated by a carriage returns, and groups of moments that formed the same movement were marked at the beginning and end with the string "XXXXXXX". An example of a recorded movement can be found in the appendix in figure 3.1.

## 3.3 Orientation & Positional Standardization:

In a virtual environment a user world ideally be able to freely navigate the virtual world, "Walk" around and explore and manipulate the environment, examining object from different viewing angles without having to worry about being positioned correctly to access a certain widget. However, if the user issues a gesture command looking straight while facing west the coordinates of the individual moments of that movement would very different than if he were to make the same gesture standing five feet further south, facing east and looking down. Asking the user to assume a specific position in order to issue a gesture command violates the concept of continuity, as described in the user interface book!

To avoid this problem all gestures go through a series of standard transformations to translate gestures into a standard position so that changes in the position and orientation of the user does not affect the values of the features extracted from the gesture. After the transformation the first sample in each gesture would have the stylus positioned at the origin (0,0,0) and the HMD positioned some negative distance -z along the z axis (0,0, -z). later samples may deviate from this starting position, but all gestures must start from here.

The first transformation translates all points so that the HMD on the first moment is on the origin (0,0,0).

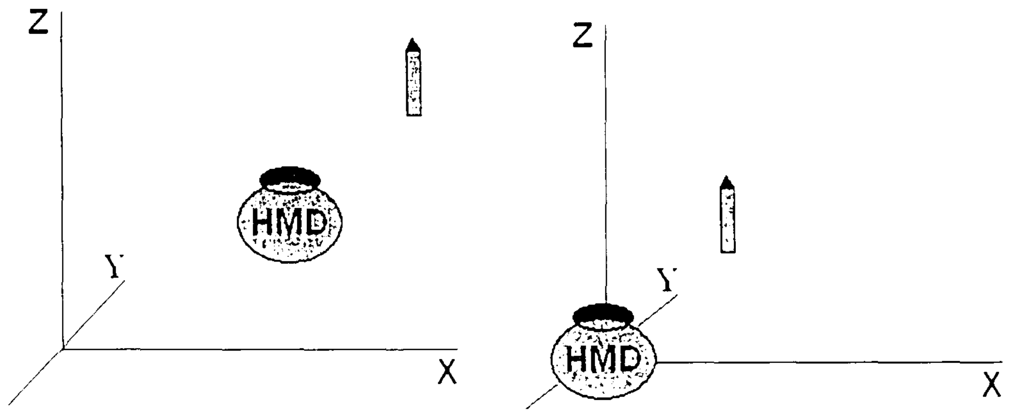


Figure 3.2

Once the HMD is on the origin, now all points are rotated along the X & Y axis so that the first moment has the stylus on the Z axis to make the X & Y values of the first moment equal to zero. Now the stylus is directly down the Z axis from the HMD.

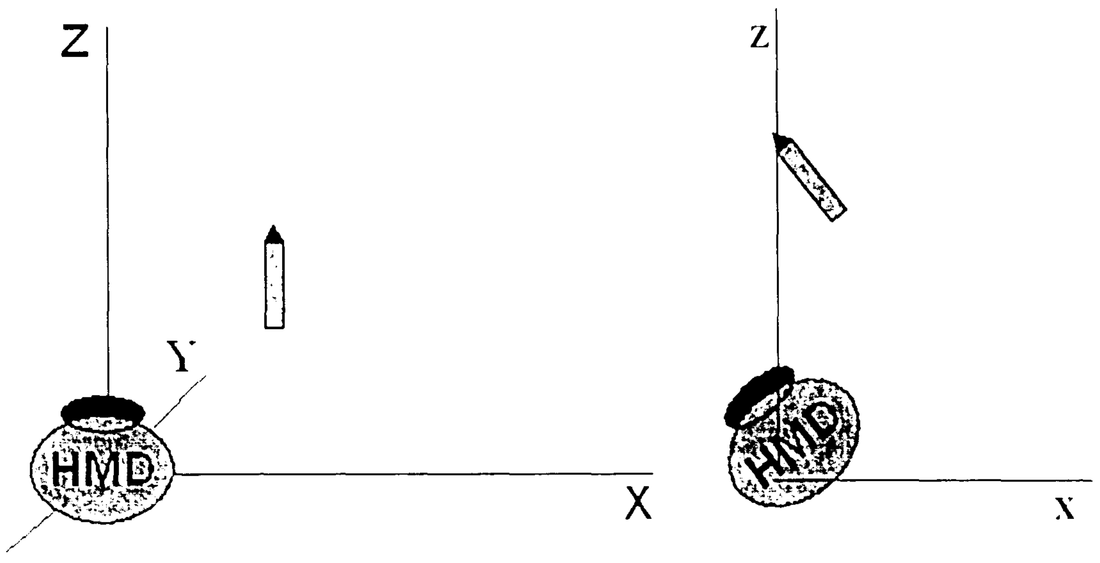


Figure 3.3

Finally, all points are translated so that the stylus on the first moment. This translation was added to the experiment later on in order to facilitate extracting a feature representing the general direction in which the gesture was made.

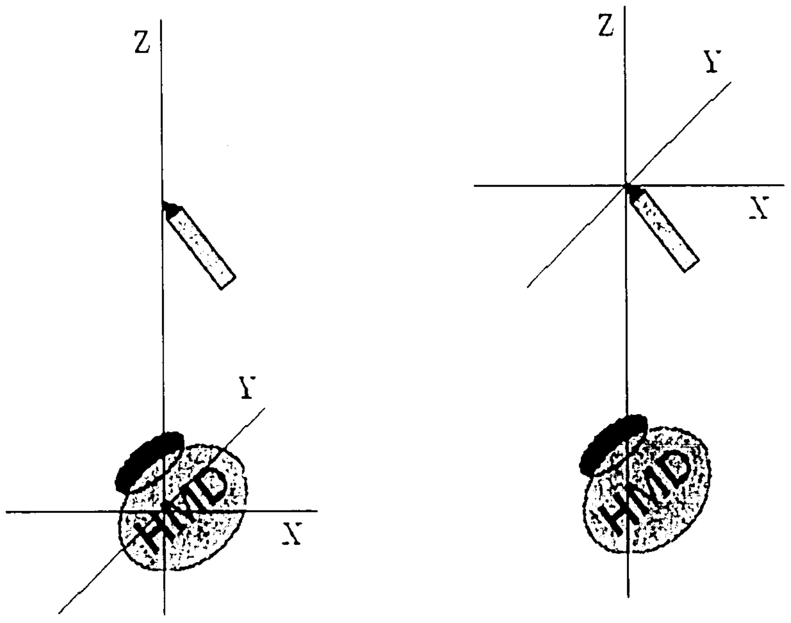


Figure 3.4

These three translations make the user's position and orientation in the virtual/physical space irrelevant when extracting features from a movement.

## 3.4 Size Normalization (Standardization)?

It has been taken into consideration as to whether the gestures should be shrunk down to a standard size. We define the size of a gesture in terms of a rectangular bounding box just big enough to contain all the points of the gesture. To find this box we view the gesture in terms of the values of the points on each individual axis. the dimensions of the box along the Z-Axis is equal to the difference between the largest and smallest value for Z between points. A similar measurement is done on the other two Axis (X & Y). Shrinking a motion requires a rigid body transformation in which first all points are translated such that the first point will be on the origin (0,0,0). then the values of all points are multiplied by the factor (Standard Size(3 space)/Current Size(3 space)) where standard size is the desired size of the motion and current size is the actual size of the motion. This is a standard size transformation in 3space. The question however is whether this transformation is useful. Its true that this eliminates variances in the size between gestures of the same class. and makes recognition a little less brittle. however this comes at a cost of losing a lot of precious information. for example losing the difference in width between the letter M and the letter I is one easy way to differentiate between the two classes. Also. if enough samples are collected and a Bayesian decision surface is employed then variances in size would be accounted for by using inverse covariance matrices in Mahalanobis distance formula instead of distance.

## 3.5 Sources of Noise

A few sources of noise have been identified in the data. The first is the varying sampling rate of the Virtual Environment hardware. Since the sampling rate is tied directly to the frame-rate, the more complicated the world becomes, the slower the frame-rate & sampling become. Even more troublesome is the difference in sampling rate within the same world. In other words the frame-rate might slow down if the users set off some functionality that is CPU intensive (i.e. engaging a complex physics engine would use up much CPU time). The timestamp associated with the sampling helps flag slowdown in frame rate, and allows the classifier to self correct, but interpolation is still going to be of lower resolution.

A second source of noise in collecting the data is the precision of the tracker. The precision of the tracker can be affected in two ways. First, because the tracker is capable of tracking an object for a fixed number of position samples per second, increasing the number of objects tracked in an application will decrease the number of times per second the stylus is tracked (120 tracks is divided equally among all the items being tracked). The tracked items in this experiment was fixed at two (head mounted display and stylus). The tracker also lost some precision the farther the tracked objects were from the tracker itself, but in this experiment (were moving around the word was not a focus of the study) distance to the tracked was kept at a reasonable

The second major source of noise is variation due to fatigue or distraction. If asked to give a large amount of sampling, fatigue in the arm and shoulder may cause faster/shorter gestures. This will vary from application to application, depending on the amount of input required from the user.

Figure 3.1 (example of a recorded movement, "Dot")

```
XXXXXXX
Xs:-0.086746 Ys:1.617598 Zs:0.020196 Xsr:-0.111504 Ysr:28.697720 Zsr:34.596798 Xh:-7.455399 Yh:1.827464 Zh:0.296726 Xhr:-
21.810287 Yhr:0.583814 Zhr:0.666028 T:312422.000000
Xs:-0.087714 Ys:1.617967 Zs:0.019477 Xsr:-0.112090 Ysr:26.060347 Zsr:33.614380 Xh:-7.473670 Yh:1.829965 Zh:0.296947 Xhr:-
21.603146 Yhr:0.374140 Zhr:0.629601 T:312484.000000
Xs:-0.090810 Ys:1.618541 Zs:0.021607 Xsr:-0.115118 Ysr:24.635727 Zsr:33.144108 Xh:-7.213874 Yh:1.831314 Zh:0.297931 Xhr:-
21.380430 Yhr:0.108264 Zhr:0.757993 T:312531.000000
Xs:-0.093496 Ys:1.621625 Zs:0.025511 Xsr:-0.119711 Ysr:25.235001 Zsr:32.614559 Xh:-6.507392 Yh:1.831561 Zh:0.299339 Xhr:-
21.238810 Yhr:-0.170624 Zhr:0.914482 T:312578.000000
Xs:-0.096046 Ys:1.625730 Zs:0.030346 Xsr:-0.125179 Ysr:26.222826 Zsr:32.340775 Xh:-6.396603 Yh:1.831183 Zh:0.300721 Xhr:-
21.096565 Yhr:-0.505695 Zhr:1.057476 T:312625.000000
Xs:-0.100104 Ys:1.629654 Zs:0.030418 Xsr:-0.131834 Ysr:27.349777 Zsr:31.699825 Xh:-6.350711 Yh:1.830873 Zh:0.302225 Xhr:-
21.023607 Yhr:-0.802461 Zhr:1.156482 T:312672.000000
Xs:-0.105816 Ys:1.628979 Zs:0.018635 Xsr:-0.139442 Ysr:26.829988 Zsr:30.600430 Xh:-6.602192 Yh:1.830935 Zh:0.303053 Xhr:-
21.098726 Yhr:-1.060452 Zhr:1.213396 T:312734.000000
Xs:-0.112107 Ys:1.621161 Zs:-0.011506 Xsr:-0.146572 Ysr:23.477976 Zsr:27.660137 Xh:-7.360325 Yh:1.831354 Zh:0.304101 Xhr:-
21.142860 Yhr:-1.270894 Zhr:1.185101 T:312781.000000
Xs:-0.115188 Ys:1.613317 Zs:-0.038199 Xsr:-0.153745 Ysr:20.443668 Zsr:24.920483 Xh:-8.537274 Yh:1.831709 Zh:0.305321 Xhr:-
21.051792 Yhr:-1.546544 Zhr:1.128377 T:312828.000000
Xs:-0.116791 Ys:1.615445 Zs:-0.038147 Xsr:-0.160505 Ysr:20.780796 Zsr:24.866531 Xh:-9.445004 Yh:1.832045 Zh:0.306052 Xhr:-
20.831671 Yhr:-1.724106 Zhr:1.096412 T:312875.000000
Xs:-0.118238 Ys:1.623124 Zs:-0.022282 Xsr:-0.167099 Ysr:22.826714 Zsr:26.042223 Xh:-10.232381 Yh:1.832362 Zh:0.305720
Xhr:-20.580038 Yhr:-1.813432 Zhr:1.141380 T:312922.000000
Xs:-0.118965 Ys:1.629931 Zs:-0.004810 Xsr:-0.171974 Ysr:25.225452 Zsr:27.033892 Xh:-10.287174 Yh:1.832461 Zh:0.304901
Xhr:-20.508881 Yhr:-1.776856 Zhr:1.248750 T:312984.000000
Xs:-0.119236 Ys:1.633510 Zs:0.003746 Xsr:-0.175248 Ysr:26.656246 Zsr:27.296761 Xh:-10.292832 Yh:1.832416 Zh:0.303944
Xhr:-20.397903 Yhr:-1.715957 Zhr:1.254155 T:313031.000000
Xs:-0.119178 Ys:1.633694 Zs:0.003711 Xsr:-0.176963 Ysr:26.865929 Zsr:27.122627 Xh:-10.575041 Yh:1.832558 Zh:0.303248
Xhr:-20.313501 Yhr:-1.758011 Zhr:1.196083 T:313078.000000
Xs:-0.119025 Ys:1.633480 Zs:0.001707 Xsr:-0.178223 Ysr:27.159090 Zsr:27.522322 Xh:-11.294308 Yh:1.832569 Zh:0.302641
Xhr:-19.990133 Yhr:-2.046072 Zhr:1.114790 T:313125.000000
XXXXXXX
```

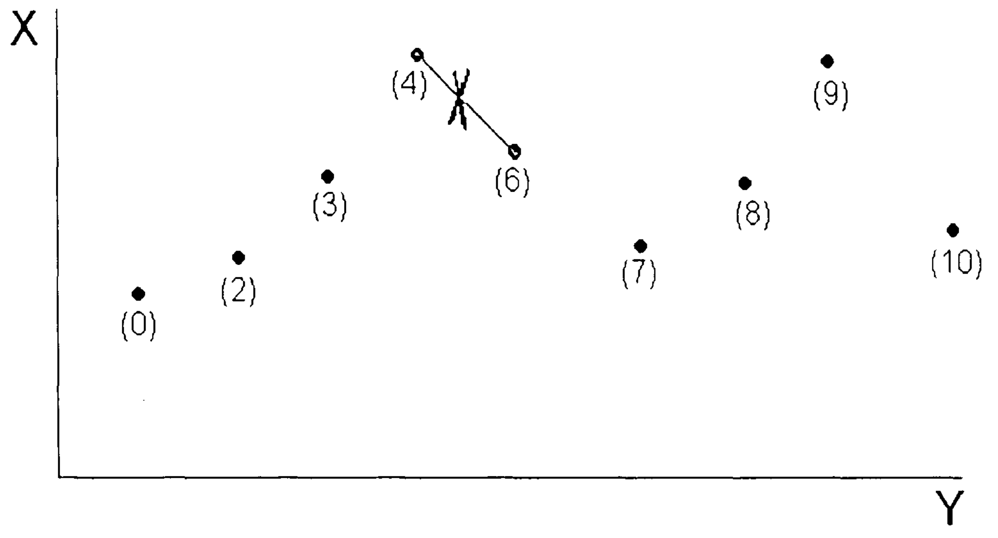
# Chapter 4

# Feature Extraction

Given a certain movement, the classifier views the motion through a set of values for certain features. Classifiers can discriminate properly between classes if gestures of the same class return features with similar values, or at least with a traceable pattern. Three sets of features were used for this project: Relative Position, Relative Direction, and Non-Form Matching

## 4.1 Relative Position

One of the differences between traditional Optical Character Recognition and what is described in this paper is that the moments come with a timestamp, which make it possible to recreate the order in which the gesture was made. this feature set takes advantage of that capability. Given a percentage the relative position feature extractor returns the X.Y.Z coordinate of the stylus at that particular phase of the gesture. for example if a classifier were to choose the relative position at 50% then the feature extractor would return the position of the stylus halfway through the gesture. Progress through the gesture is measured by time, using timestamps to calculate what the percentage of completion of the gesture is at each sampling. Since the sampling is done at every frame. it is unlikely that a sampling will be taken at exactly the moment when a position is requested. As such a simple form of extrapolation had to be implemented in order to return a logical feature value. This interpolation is a straight line connecting the positional samples. When a request for a feature arrives. the percentage is translated into a certain time relative to the start of the motion. The two samples directly before and after the requested time are found. A straight line is defined between the two points and the requested time is translated back into a percentage between the enclosing two sampling points. We move along the line segment from start to finish equal to this second percentage and return the current position.

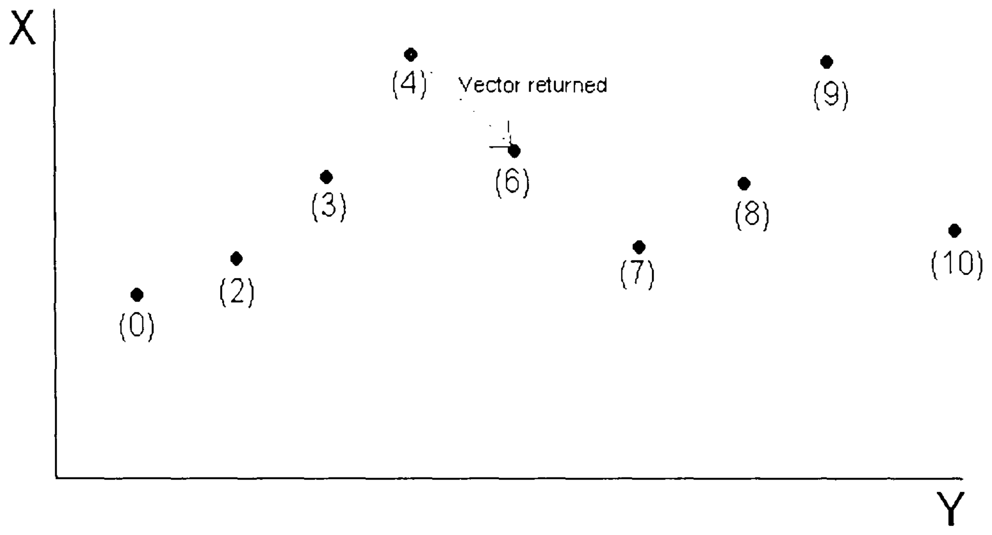


Processing the positional feature for value 50%. This would be at time 5. Time 5 is 50% down the interpolated line from from 4 to 6

Figure 4.1

## 4.2 Relative Direction

A second set of features similar to the relative position feature set is the directional sample feature (DSF) set. Like the relative position feature set the DSF takes in a percentage as an argument and returns an x,y,z value representing the vector specifying the direction which the stylus was moving in at that part of making the gesture. Just as with the RPF, the percentage is turned into a time. The two samples just before and just after the time are found and a vector connecting those two points is calculated. This vector is returned is returned as the output for that particular. Note that to maintain information on speed the vector is not normalized.



Processing Direction feature at 50% this would be at time 5
for above gesture Vector from sample at time 4 to time 6 is returned

Figure 4.2

## 4.3 Non-Form Matching

The third feature set took a different approach, and tried to look at things regardless of order, like direction changes, the bounding region and speed of the motion. A list below gives a description of the types of features extracted.

Total Distance (3 Space)

Defined as the sum of the absolute value between points in 3 space, the returned sum is the distance the stylus traveled between the beginning and the end of the gesture

Total Distance (1 Space):

Defined as the sum of the absolute value between points in 1 space, this feature is available for X, Y & Z axis separately. This is the total distance that stylus traveled on that particular axis

Number of peaks in X, Y & Z Axis:

A peak is defined as a shift in the direction of motion from positive to negative along that axis. This feature returns the number of such peaks.

Number of valleys in X, Y, Z directions:

A valley is defined as a shift in direction of motion from negative to positive along that axis. This feature returns the number of such peaks.

Ratio of distance in X, Y, Z separately to the distance in 3 space:

This returns a value of the total distance traveled in each axis divided by the total distance traveled. This returns an indication if motion was more dominant on some axis as opposed to being evenly distributed among the three.

<u>Bounding distance (1 space):</u>

This is the distance between the two farthest points on a give axis. This gives a bounding distance on that axis between which all points of the gesture would lie.

Bounding volume (3 space):

This is the minimum volume of an enclosing box (made of rectangles) required to enclose the motion. Put more simply, this is the total volume when the bounding distance in 1 space for the three axis are multiplied.

Ratio of 1 space bounding region to 3 space bounding region:

This is the 1 space bounding distance of each axis individually divided by the bounding volume (3 space). This provides a measure of if some axis had larger bounding regions or if they were generally similar in size.

Total Time:

This is the difference in the timestamp value of the first and last point sample in the gesture string.

Average speed of motion (3 space):

How fast was the stylus moving through the gesture? This is total distance (3 space) divided by total time.

Average speed of motion (1 space)

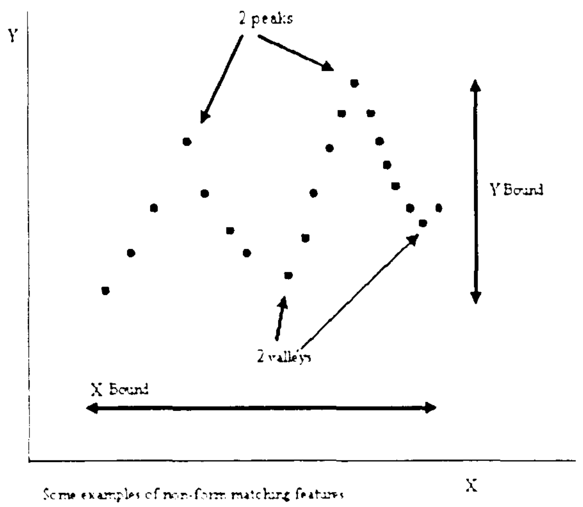
How fast the stylus was moving f only the change in a single axis is considered. Total distance (1 space) divided by Total time.

Speed at first to second and second last to last points

This feature was added as a response to an observation in the data. The target user would pause for a moment before and after making a gesture command. This feature proved helpful in differentiating gesture commands from non command motions. This feature is defined as the distance between the two points and the difference in the timestamps.

Distance first to last point (3 space)

This feature returned the distance between the first (smallest timestamp) and last point (biggest timestamp) of the gesture in 3 space.



Some examples of non-form matching features

**Special purpose features:**

The following features were added along the way often to solve specific confusions in classification.

Total x,y,z values:

This feature was very useful after the positional standardization moved the first point to the origin. These values tell something about the direction the gesture went after the first point, negative values imply more/farther motion on the negative side of the origin and vice versa.

Peaks past noise areas:

Certain gestures had many small directional changes around the origin of the motion, making counting directional changes noisy. This feature counted peaks (x,y,z directional changes) only a certain distance away from the origin, removing much of the noise. This feature was designed to help distinguish B, D & K form each other.

# Chapter 5

# Classification Methods

There were two models used for classification of the gestures: one that presented very good speed performance (Euclidean Distance to the Mean) and one which more closely captured the distribution of samples in the feature space, but at a much higher CPU cost (Bayesian quadratic classifier with individual class-conditional covariance matrices).

## 5.1 Euclidean Distance to the Mean

The first method consists of finding the average value of each feature for each class. This set of values is stored as the prototype of each class. When a sample comes in to be classified the Euclidean distance between its values and the values of the prototypes is calculated. The root mean square value of each feature across all classes is calculated and used to normalize the values, so that one feature does not overshadow another feature in scope and importance simply because of the metric chosen to measure it. The sample is then classified as the class with the smallest distance between the prototype of that class and the sample.

There are two major points to this method that prove very attractive. One is that method does not require a large amount of training samples to be recreated for a different user (keeping in mind that each user is required to provide their own specific set of training samples). The second is relatively small processor cost, which becomes more relevant given that certain modes of interaction may require this classification process to be executed once every frame. The cost of classifying a sample is roughly equal to:

(C1 x Number of classes x Number of features) + (Number of Classes X C2)

Where C1 and C2 are small constants representing the cost of a subtraction and Boolean comparison operation.

The downside to this method is it does not capture certain trends in features within a class (covariance), and certain distributions do very badly with such a method, especially bi modal distributions. As an example the distribution below shown in 2 space has two clear centers of activity at 1 and 3, but the Euclidean distance to the average method only capture that it has a mean around 2.

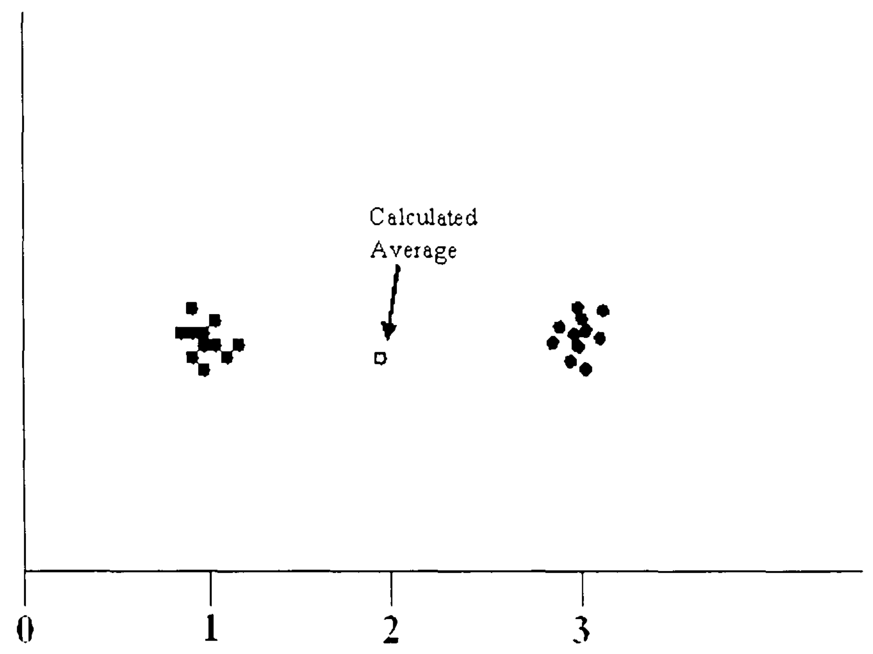


Figure 5.1

However, these problems sometimes do not come up given certain distributions. If this is the case then the Euclidean distance to the mean method is ideal. In this experiment the Euclidean Distance to the Mean method is used until failure, and then a more sophisticated Bayesian decision surface is employed.

## 5.2 Bayesian Decision Surfaces

In creating a Bayesian decision surface, the average for each sample is first calculated as it was in the step above. Once the averages are available they are used to calculate the covariance between each pair of features is calculated and stored into a covariance matrix. This matrix is then inverted and used to calculate the value of the discriminator, and helps in two ways:

First it allows the classifier to capture the true shape of a distribution rather than simply the average of all samples; skewed distributions, multimodal distributions and non contiguous decision surfaces can all be represented using a Bayesian decision surface.

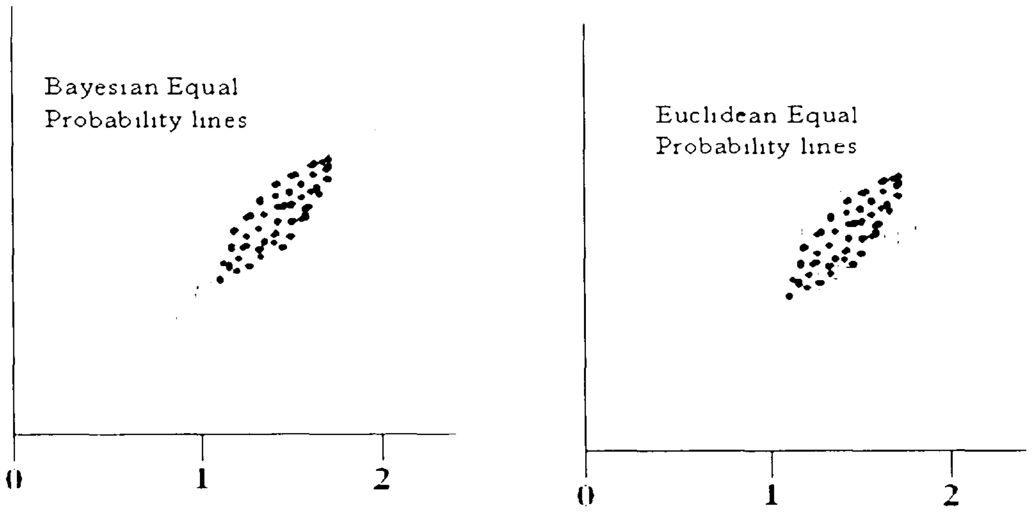
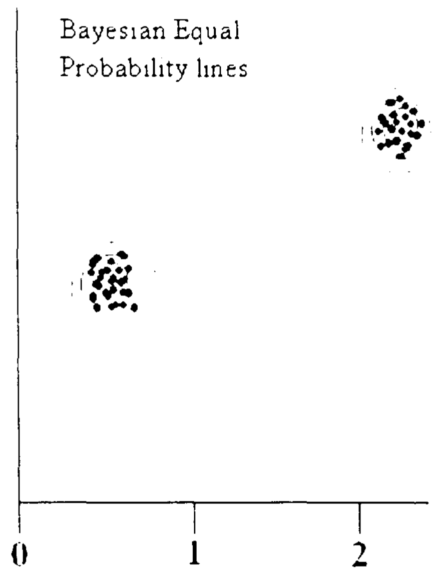
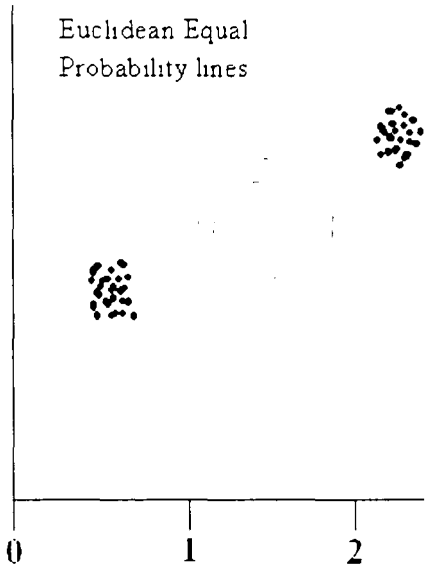


Figure 5.2

Second, by using the matrix inverse we properly weigh features that have smaller variance higher than ones with high variance, where as with the Euclidean distance classifier, all features were weighed equally.

Figure5.4


Bayesian Equal
Probability lines
0
1
2


Euclidean Equal
Probability lines
0
1
2

# Chapter 6

# Initial Experiment

An early exploratory experiment was run in order to identify the most promising of the feature sets. The feature sets that did well on a small number of classes were tested with a larger number of classes later on in the experiment.

## 6.1 The Classes

The following is a short description of the classes of gestures that were recorded and put through a classifier for this experiment. It is important to keep in mind that without a specific application's functions to model these gestures may seem somewhat arbitrary, but it is feasible to replace these gestures with different ones that are more meaningful to a specific application. Later on in this thesis when the initial experiment is complete and the data expands in scope, more universally meaningful gestures will be used. Please note that the actual gestures are 3 dimensional, the following images are 2 dimensional representations

**Quote**

This motion was meant to look like a double quote mark as it would be written on paper. This kind of gesture could be used to identify the user's intention to change modality from a sort of "free interaction" mode into a specific "Text Entry" mode and back out again.

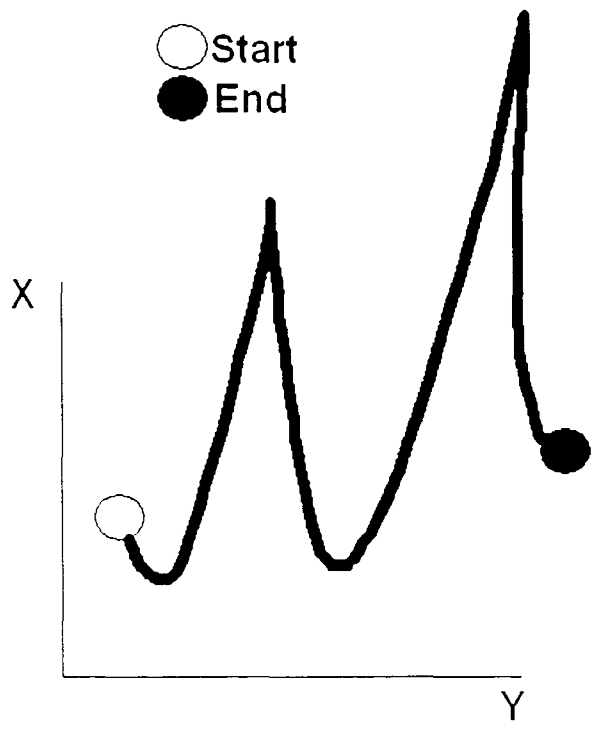


Figure 6.1

**Del**

Short of "delete", this motion was meant to look like the motion of "crossing out" some item. Indicating it is incorrect or should be deleted. This motion also looks a little bit like the character "x"

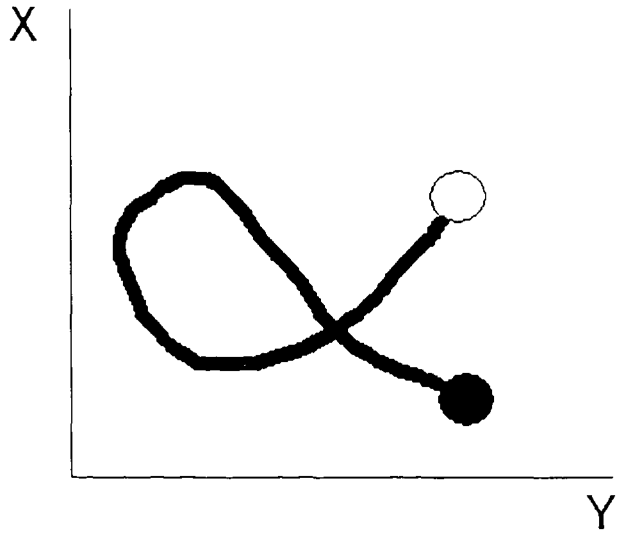


Figure 6.2

"Dot"

As suggested by the name this motion is supposed to resemble a dot. Its inclusion is interesting due to the length of the symbol (it's much shorter than the other symbols in this set). It is expected to have a smaller success rate than the other gestures because there are less sample points that the classifier can use to extract and observe features in. We will see later on that this had good and bad effects on classification. This feature also shows the ability of the VR system to track movements along the third dimension

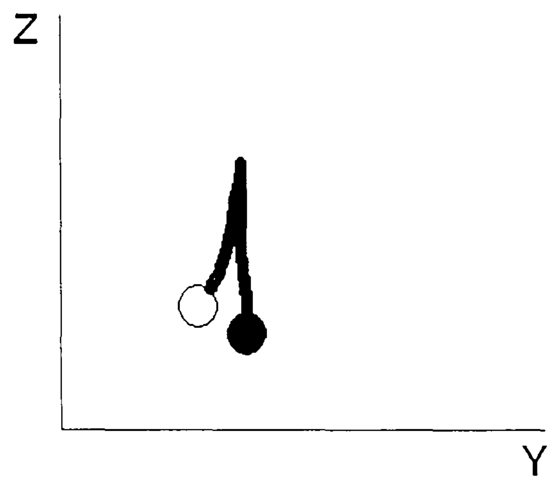


Figure 6.3

## 6.2 Relative Positional Classification

The first feature set used for classification was the Relative Position features. The results were as follows

| | Quote | Del | Dot | Errors |
|---|---|---|---|---|
| Quote | 102 | 24 | 34 | 58 |
| Del | 14 | 81 | 6 | 20 |
| Dot | 0 | 1 | 152 | 1 |
| Error | 14 | 25 | 40 | 79 |
| | | | Accuracy | 80.92% |

Figure 6.4

The feature set was not as successful as hoped in classifying the samples. The 19% error rate was indicative of certain problems in using these features for classification, especially considering that the number of classes is relatively low. It would seem that the feature set is specifically sensitive to small changes in the way the character is written. Specifically, a variation in position tends to have an accumulating effect, and as such variations early in the movement are amplified as the motion continues. In light of this early failure, this feature space was not investigated any further.

## 6.3 Relative Directional Classification

The next feature set that the Euclidean distance classifier was applied to was the Directional feature set. Recall that the DFS used the direction that the stylus was moving in at certain times. This metric did not accumulate variation, in other words, if one stroke varied slightly from one in a prototype that did not automatically cause a shift in the direction of the next stroke.

The preliminary results were as follows:

| | Quote | Del | Dot | |
|---|---|---|---|---|
| Quote | 155 | 0 | 5 | 5 |
| Del | 0 | 101 | 0 | 0 |
| Dot | 1 | 1 | 151 | 2 |
| | 1 | 1 | 5 | 7 |
| | | Accuracy | | 98.30% |

Figure 6.5

Producing an accuracy rate in the high 90's is promising, however it's important to remember that this is a small set of classes, and the results may not remain quite as impressive as the number of classes is scaled upwards to a goal of 26 characters.

## 6.4 Non-Form Matching Classification

Finally we come to test the non form matching feature set. The preliminary results were as follows:

| | Quote | Del | Dot | Error |
|---|---|---|---|---|
| Quote | 153 | 7 | 0 | 7 |
| Del | 0 | 152 | 1 | 1 |
| Dot | 0 | 4 | 97 | 4 |
| | 0 | 11 | 1 | 12 |
| | | Accuracy | | 97 10% |

Figure 6.6

Another result set with an accuracy rating in the high 90's correct classifications. Again, this is a test on a small set of classes, and accuracy is expected to drop significantly as the number of classes scale upward.

The above results were promising enough to motivate a more in-depth study on how the classifier would scale up towards a higher number of classes.

# Chapter 7

# Distinguishing Commands from Non-Commands

In previous chapters we described an interaction mode with which the application would constantly be monitoring the movements of the user, and the application was expected to recognize when the user was making a gesture command and when the user was doing something else (exploring the world, manipulating the position of virtual objects...etc). Non command gestures were recorded by turning on the coordinate recorder at times while the user performed non related tasks. These movements were then given to the classifier in the same format as the command gestures, and the classifiers ability to recognize that they were not commands was noted. Here we look at two ways of distinguishing command gestures from non-command Gestures

## 7.1 Acceptance Based Decisions

In this implementation samples of motions that were not gestures were collected. These gestures were then grouped together as a class, and treated like the other classes, i.e. the features were extracted in preprocessing, averaged out to make the prototype, then the distance from each incoming sample to the prototype of each class (including the non-gesture class) and the sample was labeled as the class whose prototype was closest to the incoming sample. Unfortunately this technique was a dismal failure, with more than half of the "none of the above" class being mistakenly classified as command gestures (false positives). The reasons for this failure are interesting for discussion because they point back the fundamental flaw of the Euclidean distance model of classification in dealing with multimodal classes and led to the development of the next Decision mechanism.

The class of all motions that are not commands is very wide, and in regards to our feature space, is for all practical reasons infinite. To demonstrate the failure of the Euclidean distance model we only need to consider to different kids of gestures. Let's say we have grouped the two actions "moving an object" and "standing still" together as two types of motions.

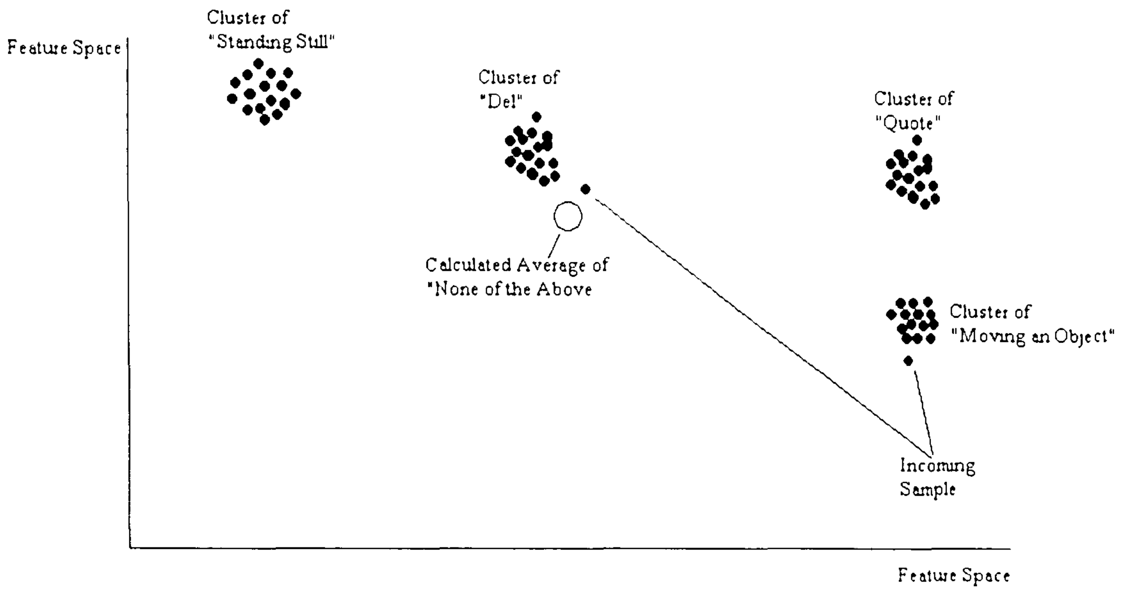

Feature Space
Cluster of "Standing Still"
Cluster of "Del!"
Cluster of "Quote"
Calculated Average of "None of the Above
Cluster of "Moving an Object"
Incoming Sample
Feature Space

Figure 7.1

Note in the above figure that when the two clusters "Standing Still" and "Moving an Object" are combined into one class their average is actually much closer to that of the "Del" cluster, and nowhere near the cluster of "Moving an Object" or Standing Still. Note what happens when the blue incoming sample is classified using the Euclidean Distance model: The sample is closest to the "None of the Above" calculated average and is classified as such, when in reality its much more probable that this sample belongs to the class "Del". This results in a false negative result, i.e. a gesture has been ignored. Now note what happens when the red incoming sample is classified. This sample most probably belongs with the cluster of "Moving an Object," but with the calculated average of "None of the Above" being so far away, this sample would be classified as a "Quote" instead, causing a false positive, i.e. a gesture command would be recognized when a user did not intend to issue a command. One way to help remedy this situation is use clustering algorithms to group together clusters and represent each one as a separate class. However, given the that number of different things a user can do that isn't a command is seemingly infinite, the number of classes would skyrocket (causing a decline in accuracy of the classifier) and on top of this the training data would have to have an enormous amount of samples, and must include every possible kind of motion, which is not a practical option for the scope of this (or any) project.

## 7.2 Acceptance Based Decisions

A different approach to classifying gestures and non-gestures is take the definition of non-command more literally: something that is not a command. In this technique we recognize that it is not necessary to note what non-command motion a particular sample looks like, but instead only that it doesn't look like any gesture commands we are specifically looking for. This technique defines a hyper-spherical volume around each class prototype. The size of this hyper sphere is dependant on the variance of distance from each training sample of that class to the prototype, i.e. this sphere will be big enough so that the vast majority the samples taken from that class are within the sphere, and each class can have a sphere of different size. The classifier first finds the class whose prototype is closest to the given sample, then classifies the sample as that class if it's within its hyper sphere. else it rejects the sample as a non-command.

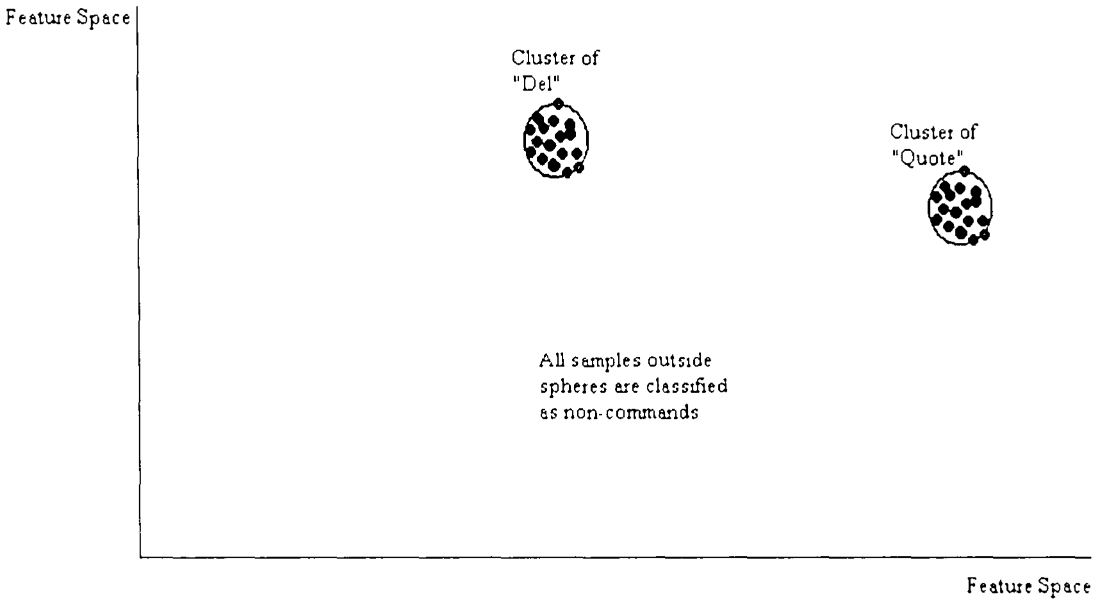


Figure 7.2

After some fine tuning. these were the results using the non form matching feature set the results were reported as follows:

Using the Directional Feature set:

| | Quotes | Del | Dot | None | Error |
|---|---|---|---|---|---|
| Quotes | 146 | 0 | 0 | 14 | 14 |
| Del | 0 | 101 | 0 | 0 | 0 |
| Dot | 1 | 0 | 120 | 32 | 33 |
| None of ab | 12 | 3 | 28 | 107 | 43 |
| error | 13 | 3 | 28 | 46 | 90 |
| | | | Error Rate | | 84.04% |

Figure 7.3

The Direction Feature Set shows an 84% accuracy rate. This is a relatively low accuracy rate. and tends to imply that the Directional Feature set could have trouble expanding the set of classes much further than three. We will explore this more later on in the paper when the number of classes is driven up

Using the Non-Form Matching feature set:

| | Quote | Del | Dot | None | Errors |
|---|---|---|---|---|---|
| Quote | 156 | 0 | 0 | 5 | 5 |
| Del | 0 | 156 | 0 | 0 | 0 |
| Dot | 0 | 0 | 137 | 16 | 16 |
| None | 3 | 1 | 7 | 137 | 11 |
| Error | 3 | 1 | 7 | 21 | 32 |
| | | | | Accuracy | 94.82% |

Figure 7.4

This approach shows a ninety five percent accuracy rate. which is rather promising given the undefined nature of the "none of the above" class. This approach solves the issue of needing to sample all the possible examples of non command. and reduces the complexity of the classifier significantly. It also gives us a new parameter to tweak which we will discuss below: tweaking the classifier towards false positives or false negatives. but before we can discuss this parameter we should take a closer look at what does each type of error entail exactly.

# Chapter 8

# Costs and Priors

This section takes a closer look at how often certain kinds of errors occur and what the cost of such errors are from the point of view of the user.

## 8.1 Priors

Looking at an error rate less than 5% seems very promising, but it in fact reflects an incomplete analysis. The table from the previous section (#) includes results from tests with similar numbers of samples from each class (about 150 samples of each class). This does not properly reflect the prior probabilities (priors) of each class, specifically, the priors of each command (dot, quote, delete for example) in a system are application dependant. In other words, each application will map a function to each gesture, and the frequency of issuing each gesture is dependant on the nature of the function. We will not restrict the mapping of functions in this paper, rather leave that flexibility for individual applications. What does remain a constant however is that in the majority of modes of interaction with an HCI is that the amount of time issuing commands is a small percentage of the total time spent interacting with the world. In the VE application the user may be manipulating objects, traversing the world or simply observing the objects within the world for ten minutes. During that time he may issue a few, ten, or say even thirty commands. If each command taking no longer on average than two second (the samples taken for this experiment rarely took longer than 1.5 seconds) then the classifier would receive two minutes worth of gesture commands, and eight minutes of non-commands. As the ratio of non-commands to command gesture increases, the total error rate will move towards the rate of false positives, giving us incentive to reduce the number of false positives, even if it were at the cost of increasing other types of errors.

## 8.2 The Cost of Errors

The three criteria used to gauge how severe the cost of an error is are

a) ***Correction***. What is entailed in correcting this error, how much effort does it take to restore the world to the state it was in before the error?

b) ***Disorientation***. Is the error easy to detect and understand by the user? Does the error cause the user to become disoriented and lose focus on the intended task?

c) ***Annoyance***. How annoying is the error to the user?

There are three cases to consider, and three categories to gauge when considering the cost to the user. The three cases are:

1. ***False Negative.*** A user issues a command and the machine classifies it as "none of the above" and does not engage any special functionality
2. ***Substitution.*** A user issues a command and the machine misclassifies the gesture as another unintended gesture (Substitution)
3. ***False Positive.*** A user is not issuing any special command, but the machine misclassifies a motion made by the user as a gesture. (False Positive)

In the following section paragraph we make the assumption that the underlying interface of the VR application which executes the functions issued has implemented the principle of visibility and blah as specified in Principles of user Interface Design.

Case 1. False Negatives is the lightest in all three categories, correcting this mistake is to simply re-issue the gesture. Since the user is waiting for a change in the state of the world, attention will be paid to the state of the world, and the absence of the change is noted (there are no unseen side-effects to a false negative), keeping disorientation to a minimum.

Case 2, Substitution could entail some effort for correction depending on the function that is un-intendedly invoked. The users attention is focused on the change in the state of the world, if the functionality properly implements the principle of visibility, then the user should be able to see that some other function was executed. The user now has to reissue the command he originally wanted.

Case 3, False Positives is perhaps the most disorienting or the three errors, since understanding why unintended functionality is being executed requires knowledge of the underlying classification system, which should not be a requirement for using an interface, especially one of more complex nature. The fact that the user's attention is not focused on the effect of the gesture recognizer on the world increases the user's disorientation. Since false positives usually imply interruption of a task that is unrelated to the gesture recognition software (observing the world, or manipulating the position of an object…etc) it would make the recognition software seem intrusive in its inaccuracy and annoy the user, who now must pick up the previous task where he was interrupted.

Of the three cases, false negative seems the most benign, while false positive seems to have the most negative effect.

## 8.3 Biasing Toward a Type of Error

The results in figureX show a classifier that is tuned to give the lowest total errors among the samples given to the classifier. However we also mentioned earlier that it is possible to bias the classifier towards false negatives or false positives. We do this by manipulating the radius of the hyper-sphere that the rejection decision is based on. Recall that the algorithm worked by finding the closest prototype to a given sample, then classifying the sample as that gesture if the sample is within the hyper-sphere defined for that prototype. As you can see in the example diagram below (X), making the hyper-sphere's radius smaller will increase the number of false positives, and decrease the number of false negatives: conversely making the sphere larger will increase the number of false negatives and reduces the false positives.

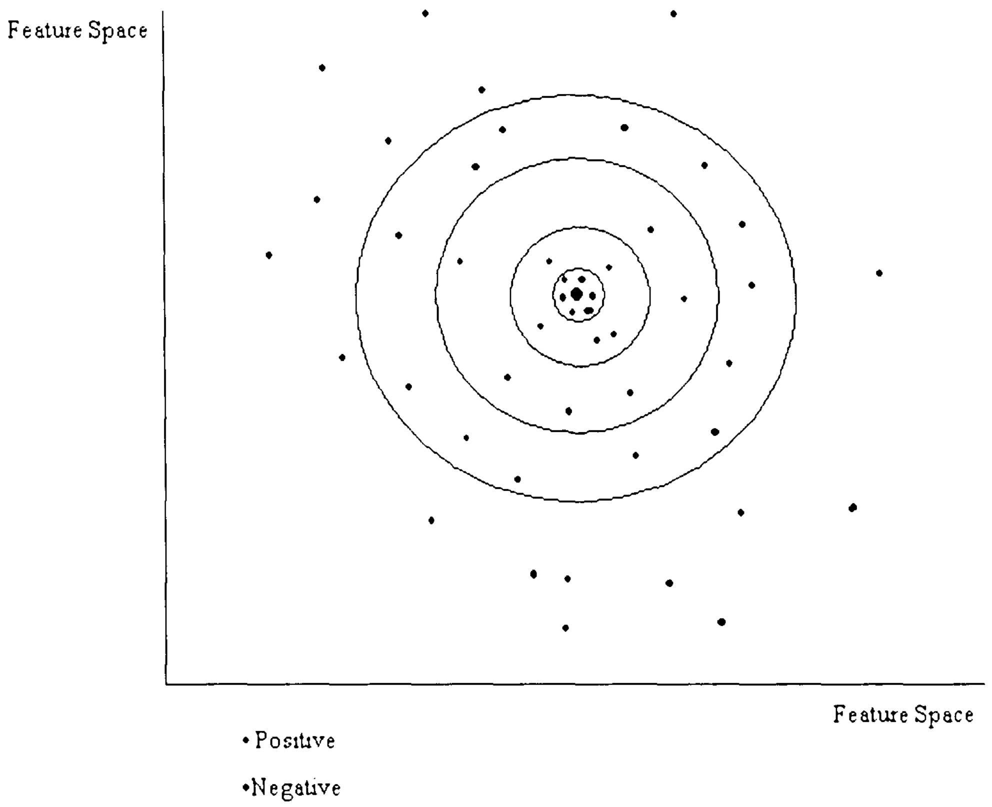


Figure 8.1

As the diagram above suggests, there is a certain radius that the hyper-sphere can reach where total error due to false positives and false negatives is at a minimum. However when the above factors of priors and cost of error suggest are weighed into the equation, it seems more beneficial to bias the classifier to make more false negatives than false positives. Even if total error becomes larger in the test samples, the cost of errors and number of total errors goes down with this biasing. The choice of the extent to which this biasing is to be taken remains application dependent, and command dependent.

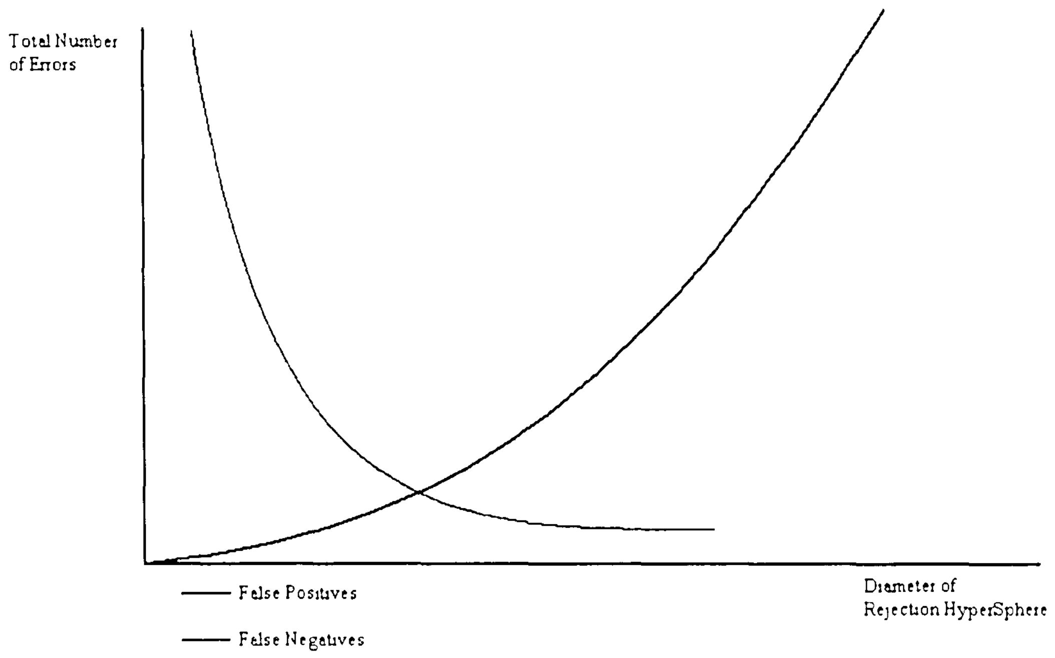


Figure 8.2

Another effect of changing the radius of the rejection hyper-sphere is on substitution errors with low confidence ratings. By shrinking the hyper-sphere we bias the classifier into rejecting samples of low confidence, moving some substitution errors into false negative.

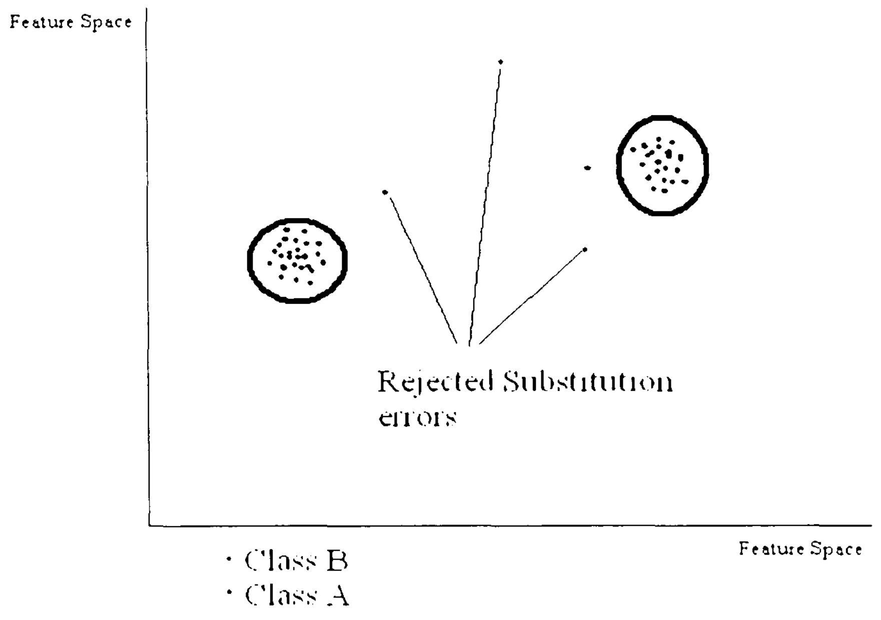


Figure 8.3 (Rejection errors that would have otherwise been substitution if they didn't fall outside the acceptance hyper-sphere)

# Chapter 9

# Expanding Towards the English Alphabet

This section reports the results of expanding the number of classes used the in the Euclidean Classifier till a gesture is present that represents each character in the English alphabet. The user provided sample gesture for all 26 characters of the English as if he were writing them in space. The gestures were then introduced to the classifier five classes at a time and the results were noted.

## 9.1 The Gestures

The user was not given any prototype to follow for writing the characters, and was only instructed to try to stay consistent in whatever representation he chose. The user described the most "natural" feeling representations for each character on paper then proceeded to provide samples in the virtual environment. The character representations are reproduced in diagram (8.1).

## 9.2 Ten Classes

The results from classifying with ten classes (A,B,C,D,E,F,G,H,I,K) using the direction sampling feature set are seen in diagram (8.2), while the non-form matching feature set results are seen in (8.3)

Both feature sets show a significant drop in accuracy, but remain above the 90% range. B & H are the worst performing classes for the non-form matching set, while G is the most erroneous class for the directional sampling feature set. We note that directional feature set is still more accurate than non-form matching set, but more importantly we notice that the two feature sets have errors distributed differently between classes, i.e. the errors seem uncorrelated. This fact can prove useful later on.

Figure 8 2 (Non Form Matching results, 10 classes)

| | A | B | C | D | E | F | G | H | I | J | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 0 | 101 | 1 | 29 | 0 | 1 | 0 | 18 | 0 | 0 | 49 |
| C | 0 | 0 | 148 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 |
| D | 0 | 11 | 0 | 133 | 2 | 1 | 1 | 2 | 0 | 0 | 17 |
| E | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 |
| F | 0 | 2 | 1 | 0 | 0 | 140 | 1 | 8 | 1 | 0 | 13 |
| G | 0 | 5 | 0 | 4 | 2 | 5 | 133 | 1 | 0 | 0 | 17 |
| H | 0 | 9 | 0 | 9 | 0 | 14 | 0 | 117 | 1 | 0 | 33 |
| I | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 1 | 147 | 0 | 3 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 |
| | 0 | 27 | 2 | 42 | 6 | 23 | 2 | 30 | 2 | 0 | 134 |
| | | | | | | | | | Accuracy | | 91 08% |

Figure 8 3 (Directional results, 10 classes)

| | A | B | C | D | E | F | G | H | I | J | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 150 | 0 | U | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 0 | 134 | 0 | 0 | 0 | 16 | 0 | 0 | 0 | 0 | 16 |
| C | 0 | 0 | 148 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 2 |
| D | 0 | U | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E | 0 | 0 | 0 | 0 | 134 | 0 | 13 | 3 | 0 | 0 | 16 |
| F | 1 | 32 | 0 | 4 | 0 | 115 | 0 | 1 | 0 | 0 | 38 |
| G | 0 | 0 | 0 | 0 | 14 | 0 | 136 | 0 | 0 | 0 | 14 |
| H | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 148 | 0 | 0 | 2 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 147 | 3 | 3 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 149 | 1 |
| | 1 | 32 | 0 | 5 | 14 | 17 | 14 | 4 | 1 | 4 | 92 |
| | | | | | | | | | Accuracy | | 93 88% |

## 9.3 Twenty Classes

The results from classifying with twenty classes (A,B,C,D,E,F,G,H,I,K, L, M,N,O,P,Q,R,S,T) can be seen in Figure 9.4 for the non-form matching set and Figure 9.5 for the directional sampling feature set. The data now shows the non-form matching set overtaking the directional sampling feature set. It was hypothesized earlier in chapter (x) that because the directional feature set did not perform very well distinguishing command gestures from non command gestures the direction sampling feature set's performance would diminish if the number of classes increased substantially. It seems we are seeing this here. The Non-form matching feature set still operates above 90%, but with obvious problem classes, such as B, H, and Q. Addressing individual problem classes will be discussed in chapter 11.

Figure 9 4(Non form-matching 20 classes)

| NONFC | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | Error |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 0 | 99 | 0 | 18 | 0 | 1 | 0 | 21 | 0 | 0 | 9 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 51 |
| C | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D | 0 | 8 | 0 | 135 | 2 | 1 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 15 |
| E | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F | 0 | 0 | 1 | 0 | 0 | 134 | 0 | 5 | 0 | 0 | 10 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 19 |
| G | 0 | 0 | 0 | 0 | 1 | 3 | 131 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 12 | 0 | 0 | 0 | 19 |
| H | 0 | 8 | 0 | 8 | 0 | 11 | 0 | 101 | 1 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 0 | 0 | 49 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 146 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 4 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 140 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 6 | 10 |
| K | 0 | 0 | 0 | 1 | 0 | 5 | 3 | 3 | 0 | 0 | 132 | 0 | 0 | 0 | 0 | 0 | 2 | 4 | 0 | 0 | 18 |
| L | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 146 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| O | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 |
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 8 | 141 | 0 | 0 | 0 | 0 | 9 |
| Q | 0 | 0 | 0 | 0 | 8 | 0 | 3 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 132 | 12 | 0 | 0 | 29 |
| R | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 12 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 3 | 121 | 0 | 0 | 19 |
| S | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 116 | 0 | 0 |
| T | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 18 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 186 | 19 |
| Error | 0 | 16 | 1 | 27 | 11 | 22 | 6 | 45 | 1 | 18 | 39 | 0 | 0 | 4 | 14 | 0 | 17 | 34 | 0 | 10 | 265 |
| | | | | | | | | | | | | | | | | | | Accuracy | | | 91 25% |

Figure 9 5 (Direction Sampling, 20 classes)

| MOTIO | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 147 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| B | 0 | 115 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 29 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 35 |
| C | 0 | 0 | 148 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 2 |
| D | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E | 0 | 0 | 0 | 0 | 112 | 0 | 11 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 23 | 0 | 38 |
| F | 0 | 12 | 0 | 2 | 0 | 32 | 0 | 0 | 0 | 0 | 36 | 0 | 1 | 0 | 0 | 0 | 0 | 70 | 0 | 0 | 121 |
| G | 0 | 0 | 0 | 0 | 12 | 0 | 130 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 2 | 0 | 20 |
| H | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 147 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 3 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 147 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 3 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 125 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 19 | 25 |
| K | 0 | 19 | 0 | 0 | 0 | 35 | 0 | 0 | 0 | 0 | 92 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 58 |
| L | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 144 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| M | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 143 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 7 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| O | 0 | 0 | 4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 105 | 25 | 0 | 0 | 0 | 14 | 44 |
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 13 | 126 | 0 | 0 | 0 | 7 | 24 |
| Q | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 160 | 0 | 0 | 0 | 1 |
| R | 0 | 1 | 0 | 0 | 0 | 19 | 0 | 0 | 0 | 0 | 25 | 0 | 0 | 0 | 0 | 0 | 0 | 95 | 0 | 0 | 45 |
| S | 0 | 0 | 5 | 1 | 27 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 77 | 0 | 39 |
| T | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 36 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 166 | 41 |
| Error | 3 | 32 | 9 | 11 | 39 | 59 | 16 | 2 | 3 | 39 | 91 | 0 | 4 | 7 | 13 | 32 | 10 | 79 | 25 | 41 | 515 |
| | | | | | | | | | | | | | | | | | | Accuracy | | | 83 93% |

## 9.4 Twenty Six Classes

The results form classifying all character gestures (A-Z) can be seen in figure 9.6 for the directional sampling feature set, and figure 9.7 there was not a very big change in accuracy results between the twenty and twenty six class test. the non-form matching set again proved more successful, running at about 90%, with the directional sampling feature set and around 80% with the directional sampling feature set.

It seems that the non-form matching set performs better at the range of 26 characters, but the fact that the errors are uncorrelated, and that the directional sampling feature set still performed better at a lower class rate can prove useful when building hybrid sequential classifiers, as is discussed in chapter 11.

Figure 9.1 (all the gestures, one for each letter)

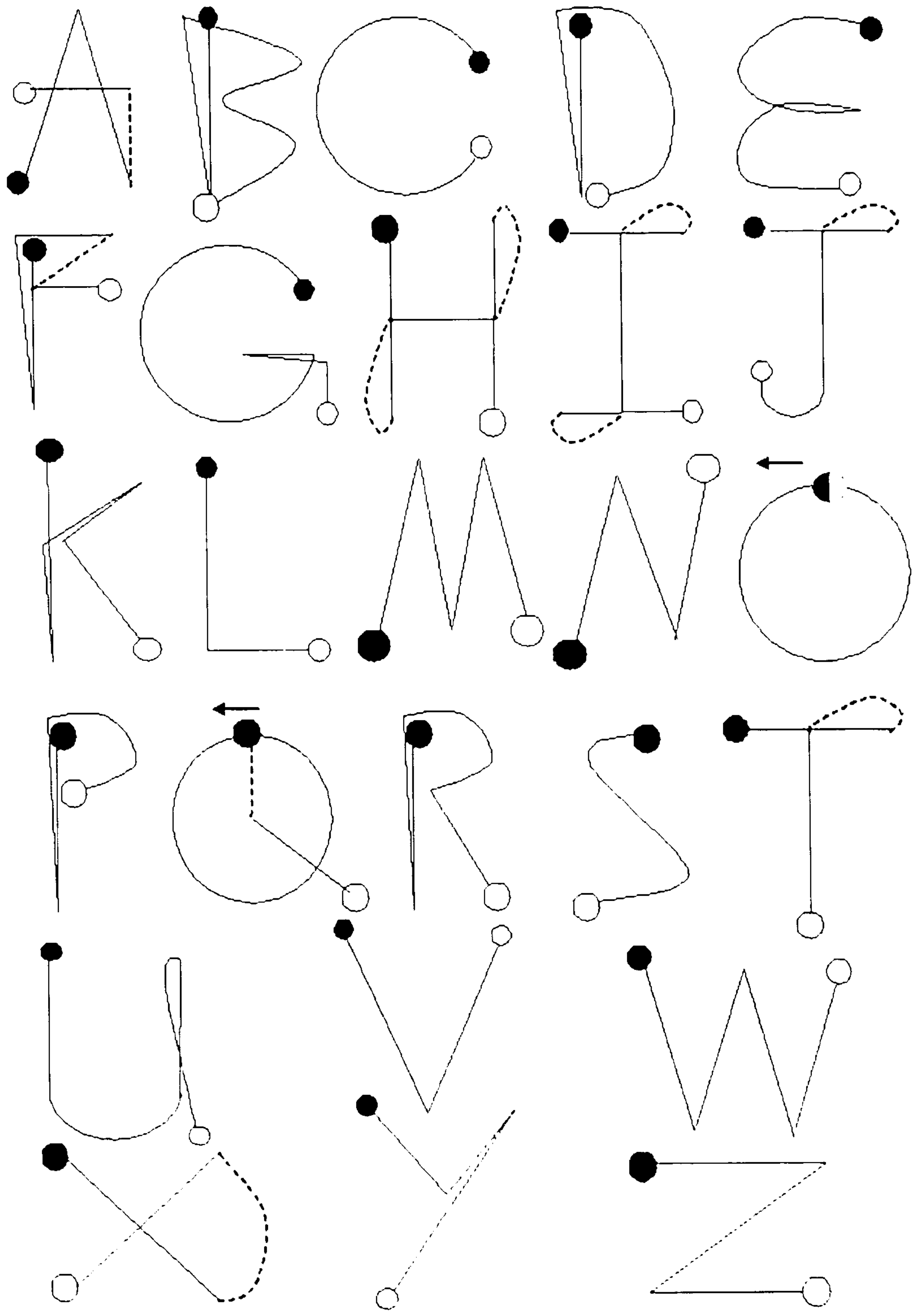


Figure 9.6 (Non Form Matching, 26 classes)

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Err |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 0 | 94 | 0 | 17 | 0 | 1 | 0 | 22 | 0 | 0 | 14 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 56 |
| C | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D | 0 | 6 | 0 | 132 | 1 | 1 | 0 | 4 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 2 | 0 | 18 |
| E | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F | 0 | 0 | 1 | 0 | 0 | 135 | 0 | 6 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 18 |
| G | 0 | 0 | 0 | 0 | 0 | 3 | 130 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 11 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 20 |
| H | 0 | 10 | 0 | 9 | 0 | 10 | 0 | 97 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 53 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 141 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 8 | 9 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 134 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 8 | 2 | 0 | 16 |
| K | 0 | 1 | 0 | 1 | 0 | 6 | 3 | 3 | 0 | 0 | 129 | 0 | 0 | 0 | 0 | 0 | 2 | 4 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 21 |
| L | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 146 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 148 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 2 |
| O | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 148 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 2 |
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 7 | 137 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 13 |
| Q | 0 | 0 | 0 | 0 | 7 | 0 | 4 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 132 | 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 29 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 16 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 120 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 20 |
| S | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 116 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 181 | 1 | 0 | 0 | 7 | 1 | 0 | 26 |
| U | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 124 | 0 | 0 | 0 | 0 | 0 | 6 |
| V | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 109 | 3 | 0 | 0 | 0 | 3 |
| W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 11 | 145 | 0 | 0 | 0 | 11 |
| X | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 137 | 5 | 0 | 13 |
| Y | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 143 | 0 | 7 |
| Z | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 158 | 1 |
| Error | 0 | 17 | 1 | 28 | 8 | 21 | 7 | 52 | 0 | 21 | 46 | 1 | 0 | 4 | 9 | 5 | 16 | 37 | 1 | 6 | 7 | 13 | 4 | 25 | 11 | 8 | 348 |

Accurracy: 91.04%

Figure 9.7 (Direction Sampling, 26 classes)

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 144 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 6 |
| B | 0 | 115 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 29 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 35 |
| C | 0 | 0 | 148 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| D | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E | 0 | 0 | 0 | 0 | 112 | 0 | 11 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 38 |
| F | 0 | 12 | 0 | 2 | 0 | 32 | 0 | 0 | 0 | 0 | 36 | 0 | 1 | 0 | 0 | 0 | 0 | 70 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 121 |
| G | 0 | 0 | 0 | 0 | 12 | 0 | 130 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 20 |
| H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 147 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 3 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 147 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 125 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 19 | 0 | 0 | 0 | 0 | 0 | 0 | 25 |
| K | 0 | 18 | 0 | 0 | 0 | 35 | 0 | 0 | 0 | 0 | 92 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 58 |
| L | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 144 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 |
| M | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 139 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 5 | 0 | 0 | 0 | 11 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 93 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 40 | 17 | 0 | 0 | 0 | 57 |
| O | 0 | 0 | 4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 98 | 23 | 0 | 0 | 0 | 13 | 0 | 0 | 0 | 1 | 0 | 10 | 52 |
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 13 | 126 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 24 |
| Q | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 159 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 |
| R | 0 | 1 | 0 | 0 | 0 | 19 | 0 | 0 | 0 | 0 | 25 | 0 | 0 | 0 | 0 | 0 | 0 | 95 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 45 |
| S | 0 | 0 | 5 | 1 | 27 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 77 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 39 |
| T | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 151 | 1 | 0 | 0 | 19 | 0 | 0 | 56 |
| U | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 130 | 0 | 0 | 0 | 0 | 0 | 0 |
| V | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 38 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 64 | 10 | 0 | 0 | 0 | 48 |
| W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10 | 137 | 0 | 0 | 0 | 19 |
| X | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 144 | 0 | 0 | 6 |
| Y | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 144 | 0 | 6 |
| Z | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 11 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 146 | 13 |
| | 5 | 31 | 9 | 11 | 39 | 59 | 15 | 2 | 3 | 39 | 91 | 0 | 10 | 47 | 24 | 30 | 8 | 79 | 25 | 42 | 7 | 55 | 32 | 22 | 0 | 10 | 695 |

Accuracy = 82.12%

# Chapter 10

# The Bayesian Classifier

## 10.1 Multi-Dimensionality and Euclidean distance

One of the issues of using a Euclidean distance classifier is that it makes each decision with no notion of how to correctly weigh a feature. When making a decision, it does not pre-compute which features are more useful than others in making a decision on which class an individual belongs to. For example: Let's assume we were trying to classify a vehicle, and for a simple example let's say we wanted to classify between "cars" and "motorcycles" based on two features: color and number of wheels. When the Euclidean distance creates the prototype of values based on some averaging technique, it would not keep track of the fact that both motorcycles and cars come in many different colors, while within the class motorcycles consistently have to wheels, and cars four. Thus a difference in color is weighed as much as a difference the number of wheels, creating much confusion in classification. In general in Euclidean space classifiers every feature is given the same weight in decision making, and the more features are added, the less significant each feature becomes.

A specific experiment of this nature was performed using the test data. Note in figure XXX that there are YY number of confusions between the character B and the character D. Also note the following diagram explaining a specific feature of the feature space. This is the "X-Peaks far from origin" feature, which counts the number of peaks on the X axis a certain distance away from the (generally noisy) origin.

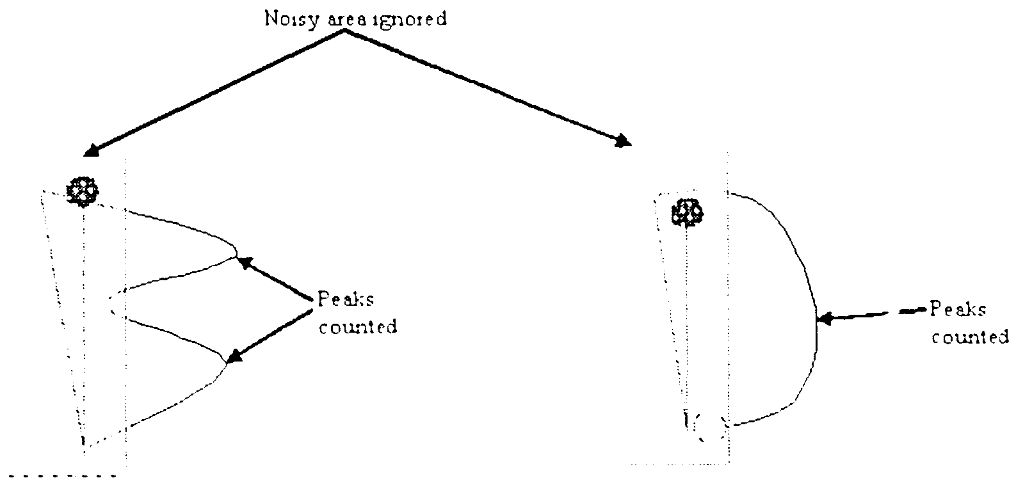


Figure 10.1

Now here are the results of a classifier based only on this feature, and only on the class B and D.

| | B | D | |
|---|---|---|---|
| B | 147 | 3 | 3 |
| D | 5 | 145 | 5 |
| | 5 | 3 | 8 |
| | | Accuracy | 97.33% |

Figure 10.2

The fact that such a simple classifier could perform that much better on this case than a 30 feature classifier promises very high potential in switching to a classifier than takes variance and covariance into account. i.e. a Bayesian classifier.

## 10.2 Results using the Bayesian Classifier

The results using the Bayesian classifier are shown in table 10.3. These results show a huge improvement over the Euclidean distance, but at a cost of much higher computation time. You will notice that the results are missing the samples of the character "W". As of now, this is an

## 10.3 Critical Flaw

The class W presented an interesting challenge that exists because of the Bayesian classifiers dependence on linear algebra. W had the unfortunate case that all samples present returned the same value for certain features (these features were integers, a good example is the feature which returned the number of valleys on the y axis. This value was consistent across all samples). Given a sample where the value is a constant is usually very helpful in identifying the class, however in this case we run into a problem of creating a row (and column) of all value zero's. Such a matrix is singular, and thus we cannot obtain an inverse matrix to continue the classification process. A solution to this (somewhat rare) problem will be investigated in the future.

## 10.4 Performance issues

Although execution time metrics were not a focus of this study, it is worth to note that the execution time of the Bayesian decision surface was much higher than that of the Euclidean distance classifier (as it was expected). The classifier used here seemed to slow to use in a real time environment, and would require some major execution time optimization before becoming a viable option in a real time application. Without optimization the complexity of the Bayesian decision classifies is of Order(Features x Features x Classes). Optimizations are out of the scope of this thesis. The next section discusses the possibility of sequential classifiers that try to improve performance

Figure 10.3 (Bayesian Non-form Matchin, 26 classes)

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F | 0 | 0 | 0 | 0 | 0 | 153 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| K | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 149 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| L | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| O | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 160 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 139 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| S | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 116 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 207 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| U | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 130 | 0 | 0 | 0 | 0 | 0 | 0 |
| V | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 112 | 0 | 0 | 0 | 0 | 0 |
| W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| X | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 | 0 |
| Y | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 150 | 0 | 0 |
| Z | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 159 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |

Accuracy = 99.92%

# Chapter 11

# Combinational Classifiers

This chapter takes a look at the possibility of constructing a sequential classifier, based on some informal experiments on the data.

## 11.1 Multiple Classifiers

Out of the feature sets that use Euclidean metric, the most successful was the non-Form matching. However, that were still specific errors that were very common.

for example B & D had a few confusions in between them. but as we showed in the previous chapter, a simpler classifier that used only two classes and one feature was more successful. But this new classifier can't be used with all 26 classes from the onset. In order to take advantage of this we would have to build a combinatorial classifier, in which certain special cases (like confusion between D & B) are identified and sent to this secondary classifier. A series of theses classifiers can form a tree which can still be faster than the (un-optimized) Bayesian classifier. Much of the groundwork of the theory for combinational classifiers has been described in detail in [3]

## 11.2 Preprocessing

The analysis required to build a sequential classifier would be done offline. There are a couple of steps that can be done iteratively to build such a classifier.

1. Identify subsets of classes that provide particularly bad confusion rates
2. Apply different classifiers to the classes. choosing the best performing and installing it as a sub classifier in the sequence for that particular subset of classes.
3. If necessary. repeat the process for the next level of classifiers until some satisfaction criteria is reached.

## 11.3 Illustrative Example

Take the case of the letter B,D & H. Note the large amount of confusion between the four classes in the Euclidean non-form matching classifier (Figure 8.x). A subset of Figure 8.x is reprinted here:

|  | B | D | H | Error |
|---|---|---|---|---|
| B | 94 | 17 | 22 | 39 |
| D | 6 | 132 | 4 | 10 |
| H | 10 | 9 | 97 | 19 |
| Error | 16 | 26 | 26 | 68 |
|  |  |  | Accuracy | 0.826087 |

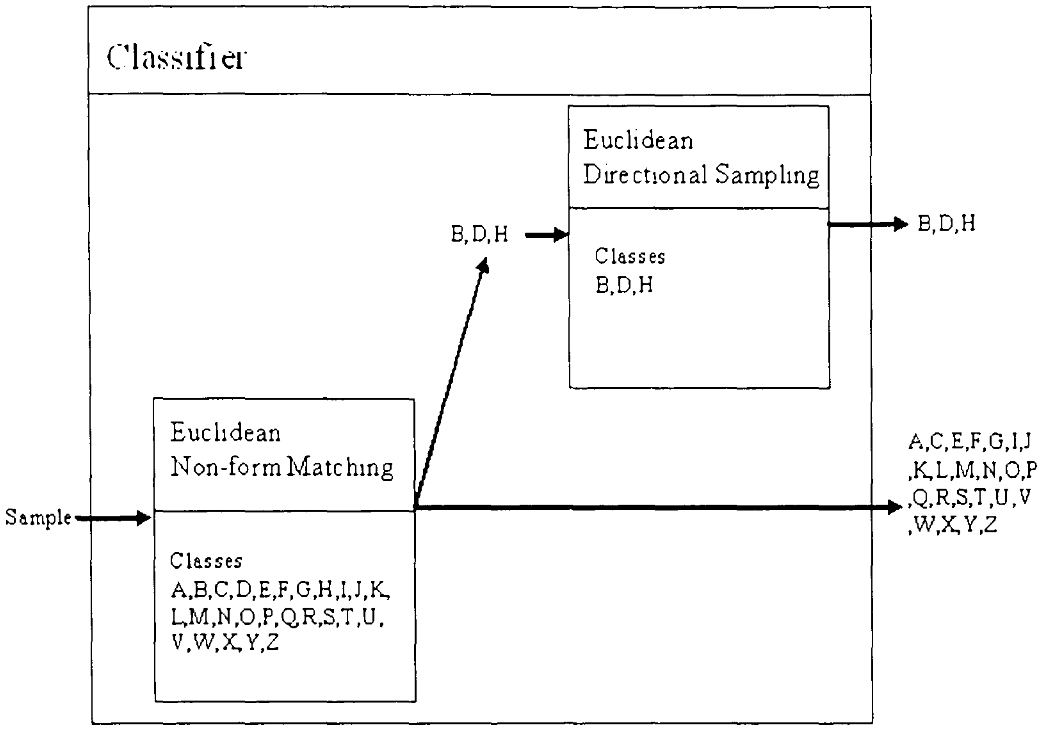
Figure 11.1

Note that the accuracy rate of this subset of letters is significantly lower than the rest of the classifier, making this a more problematic set of classes than the average. To compensate we can test this subset of classes with another classifier, in hopes that the errors are not correlated, i.e. a different classifier has better results given this set of classes. It so happens that the direction sampling feature set provides the following results if tested with only the samples for B, D & H provides the following results

|  | B | D | H | Error |
|---|---|---|---|---|
| B | 143 | 0 | 7 | 7 |
| D | 0 | 150 | 0 | 0 |
| H | 0 | 0 | 150 | 0 |
| Error | 0 | 0 | 7 | 7 |
|  |  |  | Accuracy | 0.984444 |

Figure 11.2

So a sequential classifier can decide that if the result of the first classifier returns as B, D or H, then the class is sent to a secondary classifier before making a final decision.



Classifier
Euclidean Directional Sampling
B,D,H
Classes B,D,H
B,D,H
Euclidean Non-form Matching
Sample
Classes A,B,C,D,E,F,G,H,I,J,K, L,M,N,O,P,Q,R,S,T,U, V,W,X,Y,Z
A,C,E,F,G,I,J ,K,L,M,N,O,P ,Q,R,S,T,U,V ,W,X,Y,Z

If a sample is determined to be either B,D or H, it is sent to a secondary classifier that takes a better guess between only those three classes that are highly confusing for the primary classifier

Figure 11.4

Of course, this is just an illustrative example, a more complete analysis could end in building classifiers of multiple levels, with many different types of classifiers (even classifiers that are no more than subsets of previous classifiers, in feature space or class type).

# Chapter 12

# Conclusion & Future Work

## 12.1 Conclusion

One of the obstacles facing the adoption of Virtual Environment applications is the lack of standard, effective Human Computer Interfaces. One specific concern is the lack of a keyboard, and thus the users inability to generate character strings, such as used to specify filenames, name objects…etc.

One proposed is to record the coordinates of the stylus as the user is moving it and interpret commands based on the motion of the stylus. There are many ways such an interface could be implemented. Tablet based and button based interfaces are viable options, as is the constant listener model.

Samples were collected of a small set of gestures generated by the user (the author), then a preliminary experiment was run. The preliminary success of with a small number of classes prompted an expansion of the experiment first to investigate the possibility of distinguishing intended gestures from non gestures, then expanded to classify all 26 English language characters.

Two Euclidean classifiers were implemented based on a form matching and non form matching feature set. These classifiers produced had accuracy around 80% and 90% respectively on the provided test set of 26 classes. A full Bayesian decision classifier was implemented, giving significantly better results than the Euclidean classifier, but at a significant cost in performance. Finally hybrid sequential classifiers were suggested as a solution, using different classifiers for different subsets of the data in a sequential decision making process.

## 12.2 Future work

This work centered around investigating the feasibility of using pattern matching techniques. The work was centered on one user (the author of this paper) due to time restrictions, and lack of permission to run experiments on human subjects. A next step in the study would be a formal investigation of expanding the user base significantly, including users with varying levels of expertise in Virtual Environment applications.

In general, the results from this experiment were promising, but they leave room for improvement. Further investigation into optimizing the classification techniques and inventing better, more descriptive feature sets would be an interesting avenue to pursue.

One big area for future work is segmentation of between a string of gestures. This experiment relied on manual intervention by the user to mark the beginning and end of a movement. The fluidity of the system, and the speed at which it can accept gestures would benefit greatly if no manual intervention was required, or at least reduced to a very fast, intuitive motion.

For this experiment, the primary user provided a large number of training samples, a number of samples on the order of 5000 or so were used to create the data used for this experiment. The large number of samples necessary to recreate this for a second user is not acceptable. An investigation needs to be carried out as to how to significantly reduce the number of training samples necessary, either by some sample amplification technique or classification methods that require less samples. Another possibility is to investigate the use of a more global set of training samples, i.e. not requiring each user to provide his own set of training samples, or use an incremental learning style where only some of the classes require new training data.

A specific problem that is yet to be addressed is the inability of the linear algebra to deal with the case of the singular matrix. Without the ability to create an inverse matrix, the classifier cannot continue. for this experiment changing the original matrix in such a way that it doesn't lose a lot of descriptive power but become non-singular was unfortunately not successful. Work will be done to address the issue in this specific case, and maybe in the general case.

Algorithms for automatic or semi automatic configuration of hybrid sequential classifiers should also be investigated and implemented. These algorithms would follow the steps described loosely in Chapter 11.

These obstacles, some of which address more general challenges in the pattern matching field, present fruitful directions for future projects and research.

## Bibliography

[1] Frees. Scott. Khouri. Rami H. Kessler. G Drew. "Connecting the Dots: Simple Text Input in Immersive Environments," *vr*, pp. 265-268, IEEE Virtual Reality Conference (VR 2006), 2006.

[2] Norman, Donald. The Design of Everyday Things, Doubleday, 1988

[3] Ho. Tin Kam. A Theory of Multiple Classifier Systems and its application to Visual Word Recognition. Ph.D. dissertation. State University of New York at Buffalo. 1992

[4] Bowman. D., Wingrave. C., Campbell. J., & Ly. V. (2001).
Using finch Gloves for both Natural and Abstract Interaction Techniques in Virtual Environments. *Proceedings of HCI Imitational*. New Orleans. Louisiana.

[5] Bowman. D., Rhoton. C., and Pinho. (2002) M. Text Input Techniques for Immersive Virtual Environments: an Empirical Comparison. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. pp. 2154-2158

[6] Fels. S., & Hinton. G. (1998). Glove-TalkII: A Neural Network Interface which Maps Gestures to Parallel Formant Speech Synthesizer Controls. *IEEE Transactions on Neural Networks*. 9(1). 205-2 12.

[7] Lindeman. R., Sibert. J. Hahn. J., (1999). Towards Usable VR:
An Empirical Study of User Interfaces for Immersive Virtual Environments. *Proc Of the SIGCHI '99*. pp.64-71

[8] Matias, E., MacKenzie, I., & Buxton, W. (1993). Half- QWERTY; A One-handed Keyboard Facilitating Skill Transfer from QWERTY. *Proceedings of ACM INTERCHI*, 88-94.

[9] Poupyrev, I., Tomokazu, N., & Weghorst, S. (1998). Virtual Notepad: handwriting in immersive VR. *Proceedings of the IEEE Virtual Reality Annual International Symposium*, 126-132.

[10] Rosenberg, R., Slater, M. (1999) A Chording Glove: A Glove- Based Text Input Device. *IEEE Transactions on Systems, Man, and Cybernetics*. Vol 29, No.2.

[11] Thomas, B., Tyerman, S., & Grimmer, K. (1998). Evaluation of Text Input Mechanisms for Wearable Computers. *Virtual Reality: Research, Development and Applications*. 3, 187-199.

[12] Zhai, S., Hunter, M., & Smith, B. (2000). The Metropolis Keyboard - an Exploration of Quantitative Techniques for Virtual Keyboard Design. *Proceedings of the ACM Symposium on User Interface Software and Technology*, 119-128.

[13] Pavlovic, Vladimir Ivan. Dynamic Bayesian Networks for Information Fusion With Application to Human Computer Interfaces. Ph. D. Dissertation, University of Illinois at Urbana-Champaign, 1999

[14] Bowman, D. A. and Hodges, L. F. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In 1997 Symposium on Interactive 3D Graphics, pages 35--38.

Rami Khouri

Date of Birth: 06/24/1983
Location: Doha, Qatar
Mother: Siham Khouri
Father: Hani Khouri

Institutions Attended:

Parkland High School (1996 – 2000)
Allentown PA.

Penn State University (2000 – 2002)
Fogelsville PA.

Lehigh University (2002 – 2006)
Bethlehem PA.

Degrees:

High School Diploma (June 2000)
Parkland High School, Allentown PA.

B.S. in Computer Science. (May 2005)
Lehigh University, Bethlehem PA.

Candidate for
M.S. in Computer Science (May 2006)
Lehigh University, Bethlehem PA.

END OF
TITLE