TO DRAW OR NOT TO DRAW: RECOGNIZING STROKE-HOVER INTENT IN GESTURE-FREE BARE HAND MID-AIR DRAWING TASKS

A Thesis

by

UMEMA HAKIMUDDIN BOHARI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Vinayak
Committee Members,	Douglas Allaire
	Eric Ragan
Head of Department,	Andres Polycarpou

August 2018

Major Subject: Mechanical Engineering

ABSTRACT

Over the past several decades, technological advancements have introduced new modes of communication with the computers, introducing a shift from traditional mouse and keyboard interfaces. While touch based interactions are abundantly being used today, latest developments in computer vision, body tracking stereo cameras, and augmented and virtual reality have now enabled communicating with the computers using spatial input in the physical 3D space. These techniques are now being integrated into several design critical tasks like sketching, modeling, etc. through sophisticated methodologies and use of specialized instrumented devices. One of the prime challenges in design research is to make this spatial interaction with the computer as intuitive as possible for the users.

Drawing curves in mid-air with fingers, is a fundamental task with applications to 3D sketching, geometric modeling, handwriting recognition, and authentication. Sketching in general, is a crucial mode for effective idea communication between designers. Mid-air curve input is typically accomplished through instrumented controllers, specific hand postures, or pre-defined hand gestures, in presence of depth and motion sensing cameras. The user may use any of these modalities to express the intention to start or stop sketching. However, apart from suffering with issues like lack of robustness, the use of such gestures, specific postures, or the necessity of instrumented controllers for design specific tasks further result in an additional cognitive load on the user.

To address the problems associated with different mid-air curve input modalities, the presented research discusses the design, development, and evaluation of data driven models for intent recognition in non-instrumented, gesture-free, bare-hand mid-air drawing tasks.

The research is motivated by a behavioral study that demonstrates the need for such an approach due to the lack of robustness and intuitiveness while using hand postures and instrumented devices. The main objective is to study how users move during mid-air sketching, develop qualitative insights regarding such movements, and consequently implement a computational approach to determine when the user intends to draw in mid-air without the use of an explicit mechanism (such as an instrumented controller or a specified hand-posture). By recording the user's hand trajectory, the idea is to simply classify this point as either hover or stroke. The resulting model allows for the classification of points on the user's spatial trajectory.

Drawing inspiration from the way users sketch in mid-air, this research first specifies the necessity for an alternate approach for processing bare hand mid-air curves in a continuous fashion. Further, this research presents a novel drawing intent recognition work flow for every recorded drawing point, using three different approaches. We begin with recording mid-air drawing data and developing a classification model based on the extracted geometric properties of the recorded data. The main goal behind developing this model is to identify drawing intent from critical geometric and temporal features. In the second approach, we explore the variations in prediction quality of the model by improving the dimensionality of data used as mid-air curve input. Finally, in the third approach, we seek to understand the drawing intention from mid-air curves using sophisticated dimensionality reduction neural networks such as autoencoders. Finally, the broad level implications of this research are discussed, with potential development areas in the design and research of mid-air interactions.

DEDICATION

Dedicated to:

My parents: Atiya and Hakimuddin Bohari -

For kindling and nourishing that tiny spark which could have been very easily lost...I owe you every bit of what I am today.

Murtuza Bohari -

Baby brother - all those times when the muggles were trying to bring me down, you have been the Patronus that I could never conjure myself. I owe you my sanity.

Varsha Joshi & Alka Kamble -

The friends I need but never deserved: For sticking by my side through the toughest of times, and for not giving up on me when all I wanted to do was give up.

Mustafa Lokhandwala -

For patiently handling my madness, idiosyncrasies, workaholic bouts, carelessness, and all downright crazy shenanigans over the past several months - I owe you one.

ACKNOWLEDGMENTS

Pursuing research is a challenging task. From the tiny bit that I've experienced, research typically involves an endless cycle of making, breaking, rediscovering yourself, and then, throwing everything back to the flames – only to start again the next day: all day, every day. This would not have been possible at all without the tremendous knowledge, guidance, support, and motivation that I received from my advisor and mentor, Dr. Vinayak. I express my sincere gratitude towards him for being extremely patient, understanding, and for teaching me the subtle art of nurturing and fulfilling king-size dreams.

I would like to thank my committee members, Dr. Douglas Allaire and Dr. Eric Ragan, for agreeing to be on my committee, and for their valuable feedback and guidance throughout. Through two of the most important courses I studied during my graduate studies, Dr. Allaire taught me that there is a way to explain critical and complicated things through simple, basic fundamentals - and I am grateful towards him for teaching me that. Dr. Ragan on the other hand, helped me understand the very core fundamentals behind a good virtual reality based interface, and how to accommodate users' preferences while designing one. I am grateful to Dr. Yu Ding – his brilliant teaching methodologies and absolute mastery over a subject as abstract as machine learning and analysis helped me get a better understanding of lots of fundamentals that have been used while writing this thesis.

I owe a big thank you to graduate researchers and my fellow lab colleagues Ronak Mohanty, Ting-Ju Chen, and Peng Jiang for their constant support all through the time I spent at Mixed-Initiative Design Lab. Your technical and innovative insights served as the most natural solutions to my research problems. A special mention to Ting-Ju – my first collaborator for this thesis – and her artistic, ingenious insights that helped me fix a lot of sub-standard Powerpoint presentations. I would also like to thank undergraduate researchers July R. Garcia and Ryan Alli for their help through the data collection studies.

Friends who have been more than a family to me at A&M : Srishti Jain – I don't know how

I'd have done any of this if you'd not have been around. Thank you for handling me at my lows - which were quite abundant over the last two years. Aryan Mehta – thank you for teaching me that life is not meant to be taken this seriously - it all works out in the end. Sukhada Saoji – I found a friend in you when I least expected or wanted to - thank you for being the most understanding room mate ever. Finally, I am grateful towards my family for all their support all through these years, and to Leonardo, the Hamster – for staying up all those nights with me when I wrote this.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Dr. Vinayak and Dr. Douglas Allaire of the Department of Mechanical Engineering, and Dr. Eric Ragan of the Department of Architecture and Visualization.

Graduate student researcher Ting-Ju Chen, and undergraduate student researchers July Garcia and Ryan Alli provided invaluable contributions during conducting and managing both data collection studies described in Chapters 3 and 4.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

The graduate student was supported through a Graduate Research Assistantship as a part of start-up fund by the Texas A&M University and Department of Mechanical Engineering.

The graduate student was also party supported by the Emil Beuhler Graduate Student Fellowship by the Department of Mechanical Engineering, and partially by AggieChallenge 2017.

The graduate study was partially supported by the D. K. Bhave Scholarship granted by the Savitribai Phule Pune University, India.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

Over the years, several advances in technology have resulted in a rising interest towards developing user centric applications. Shifting from user-driven applications where the user or designer herself is the sole contributor in performing a certain task and the other agent (typically, a computer) being a mere enabler, there has been a growing trend towards exploring *mixed-initiative* interactions and applications. Quoting Marti Hearst [1]: "*Mixed-Initiative refers to a flexible interaction strategy, where each agent can contribute to the task what it does best*". In other words, the notion of mixed-initiative refers to a meaningful communication between two agents (say, a human and a computer), where either of the two can take an initiative towards performing an action that helps towards achieving a common goal.

To that effect, at a minimum, a mixed-initiative interface should include the following basic components: (a) a mode of communication between the two agents, (b) recording and deciphering the input received from each agent, and (c) a logical, well-structured initiative from either or both involved agents to perform certain activities. Each of these components have received sufficient attention from the design research community, and with the current state of art, techniques now exist that establish an effective pipeline of information flow through traditional modalities like mouse-click interactions, and touch based input. With the immense developments in computer vision, different vision based modalities such as skeleton tracking cameras, depth sensing devices, and wearable controllers have paved the way for additional modes to achieve spatial interactions with the computer.

The prime motivation looks at the bigger picture of moving towards mixed-initiative interfaces where the designer interacts with the computer using bare hands. To decipher the spatial input received from the designer in a as-natural-as-possible setting, it is first necessary to understand the designer's intention. Strictly in the context of mid-air drawing using bare hands, the primary aim of this thesis is to explore ways in which the designer's intention to draw can be captured through models derived from bare hand movements in the physical space.

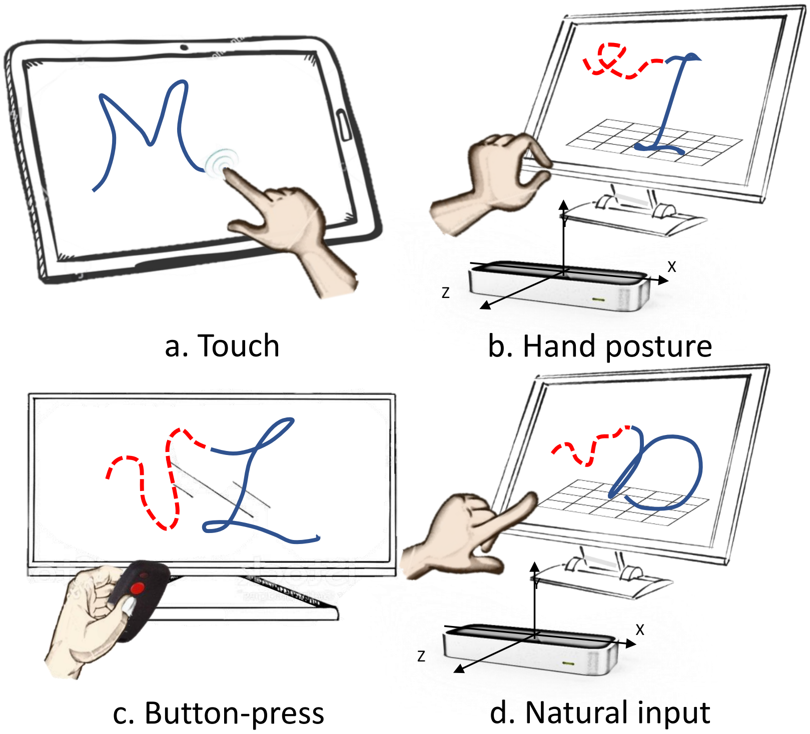
1.1 The Context of Mid-Air Curve Input

The last few decades have witnessed a steady growth in different modes of communication with the computer using hand-arm movements. This form of communication in mid-air or free 3D space, is a fundamental task of mid-air interactions with applications towards 3D sketching [2, 3], geometric modeling [4], hand-writing [5, 6, 7], and spatial authentication [8]. Recently, there has been significant interest in techniques for recognizing symbols within curves drawn in the air [6, 9].

In context of 3D sketching and modeling, these techniques process the curve input to identify *what* is being drawn, i.e. determining the semantic content of the curve input. Moreover, these techniques are not always scalable to spatial inputs due to the additional uncertainty added by the third dimension: the intended planarity of strokes is not guaranteed.

When sketching on a desktop or a tablet, the distinction between a stroke (*what the user actually intends to draw*) and a hover (*all other movements that are not intentional to the drawing task*) is trivially accomplished through explicit events such as mouse button down or touch down (Figure 1.1(a)). In mid-air interaction, such explicit *interrupts* can either be provided with some specific hand posture (Figure 1.1(b)) or with a hand-held device with buttons (Figure 1.1(c)). Most existing approaches embody this requirement in their hardware setup and use events (such as pen up, pen down, hand posture, etc.) that segment the hover points from stroke points for performing further analysis for the stroke segments of the curve.

Mid-air curve input combined with interaction devices such as Wii Remote, skeletal and hand tracking cameras like Microsoft's Kinect and Leap Motion, etc. enable applications such as gesture recognition [9, 10, 11, 12], hand-writing [5, 6, 7], and sketching [2, 4, 3]. In conventional gesture recognition tasks, the mid-air curve input is typically matched against pre-defined templates to identify the user's intention. Activity recognition tasks use specifically designed wearables that monitor the user's movements and vital stats, and use sophisticated template matching and time-series data processing algorithms to identify what activity is being done. Despite significant advances, these vision based techniques however face challenges related to tracking robustness, hand



a. Touch

b. Hand posture

c. Button-press

d. Natural input

Figure 1.1: Different mid-air curve input modalities.

pose estimation, and ubiquity of these special devices.

There are three observations in relation to the previous works that motivate the problem and approach. First, as Taranta et al. [9] note, the particular effectiveness of such recognizers for segmentation and recognition of curve inputs on touch surfaces have not been particularly successful in higher dimensional spaces. Second, in order to expand the scope of intelligent user interfaces beyond symbol recognition to free-form design interfaces, there is a need for methods that do not rely on comparing user input with canonical shapes in a repository. Most techniques are developed and evaluated using segmented data, rather than considering a continuous time-stamped data sequence of 3D points [13, 14]. To address this issue, Krishnan et al. [15] develop and evaluate a sliding window based approach to perform activity recognition with streaming sensor data. The approach adopted in JackKnife [9] proposes implementation of dynamic time warping for continuous dynamic identification of gestures. While JackKnife [9] treats each gesture as a time series representation of direction vectors and then classifies with regard to all templates stored in a database, our work processes mid-air curve input as sequential points which have individual set of feature vectors, and performs the classification on each point as and when new points are recorded. Finally, unlike sketching on a tablet with a finger or stylus, using a spatial device or a hand posture is not necessarily natural [4]. Works such as Data Miming [16] and grasp-based virtual pottery [17] have demonstrated that for continuous and free-form tasks such as design, users' movements are guided by their interactions with the physical world rather than actions prescribed by the interface designer. We draw from these latter approaches and present a method that encapsulates human movement patterns during mid-air sketching within an intelligent framework for sketch intent recognition.

The primary goal of this research is to investigate a complementary yet fundamental problem in mid-air sketching task: determining when users actually intend to draw in mid-air. On a broader level, the following questions motivate this research:

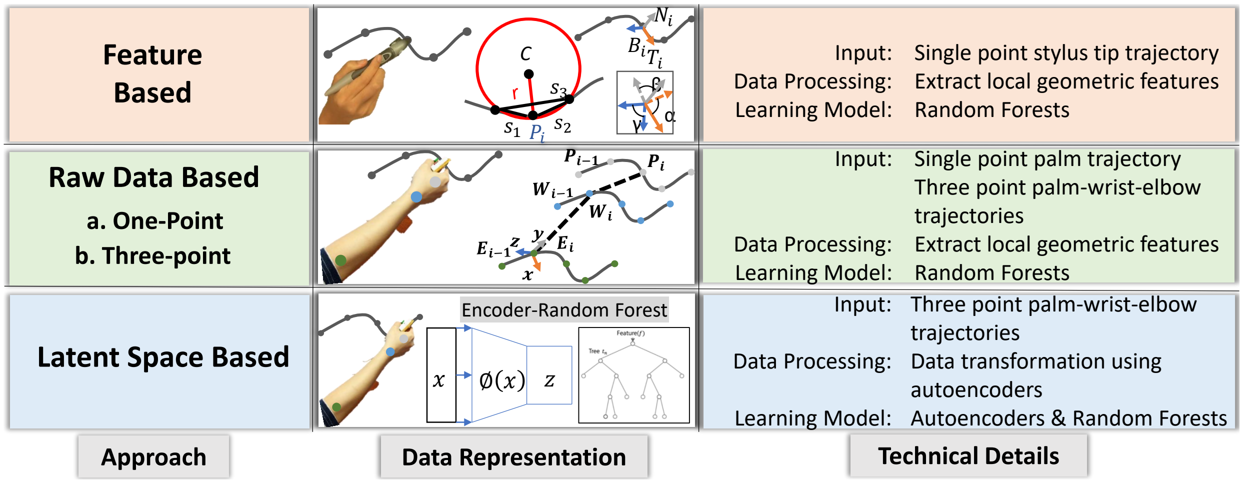
1. How do users draw in mid-air to express general shapes? What is the effect of spatial input devices or hand postures on the intuitiveness of mid-air sketch input?

2. For processing and recognition of symbols and alpha-numeric patterns, a library of predefined canonical shapes is needed. How do we approach recognition of sketches that represent known objects but may not be composed of individually recognizable strokes (such as design sketches)?

3. Does the movement of the hand while drawing in mid-air posses an inherent geometric and temporal structure that can be used to identify when the user intends to draw in mid-air?

4. Can this behavioral model be scaled to intent recognition for other activities encountered in different design tasks?

Motivated by these questions, this research aims towards studying how users move during mid-air sketching, develop qualitative insights regarding such movements, and consequently implement a computational approach to determine when the user intends to draw in mid-air without the use of an explicit mechanism (such as an instrumented controller or a specified hand-posture). This research aims at enabling future interfaces that will be able to eliminate the need for a prescribed set of gestures/postures or controllers to allow users to express 3D artifacts.



Approach	Data Representation	Technical Details
Feature Based		Input: Single point stylus tip trajectory Data Processing: Extract local geometric features Learning Model: Random Forests
Raw Data Based a. One-Point b. Three-point		Input: Single point palm trajectory Three point palm-wrist-elbow trajectories Data Processing: Extract local geometric features Learning Model: Random Forests
Latent Space Based	Encoder-Random Forest	Input: Three point palm-wrist-elbow trajectories Data Processing: Data transformation using autoencoders Learning Model: Autoencoders & Random Forests

Figure 1.2: Feature-based, raw data based, and latent space based stroke-hover classification approaches discussed in this thesis.

1.2 Contributions

In context of using mid-air curve input for 3D drawing tasks, the presented research makes the following contributions. First, an observational study characterizing hand trajectories generated by users in mid-air sketching tasks with three interfaces is presented. This study provides a better understanding of the quantitative aspects of spatial user input in terms of relative speeds of stroke and hover, intuitiveness of postures and instrumented controllers for sketching, and the types of shapes such as characters, shape primitives, and general multi-stroke shapes. Second, and the primary contribution of this research, is presenting a novel approach of processing mid-air curve input as an intent recognition task, instead of a curve segmentation process. The work looks at detecting the person's intention to draw using mid-air curve input as a non-segmentation problem. Complementary to existing approaches which look at the problem as a segmentation task, and perform all processing after the entire curve is recorded, the proposed approach treats the curve input on an on-the-fly point-by-point basis. While other works wish to identify what shape is being drawn, the presented research first asks the question of *whether anything is being drawn or not*. Which essentially makes it a point-to-point intent classification problem (*stroke* or *hover*).

Third, the research presents a data driven approach to computationally determine the *stroke-hover* intent from user recorded hand trajectory data without using postures or controllers. Given a sequence of points in the user's finger trajectory, a binary classifier is trained to learn the relationship between the motion profile and geometric features of each point in the trajectory with its true classification (hover or stroke). The resulting model allows for the classification of points on the user's spatial trajectory. To present a computable understanding of how people move while drawing in air, a data collection study is first conducted using the hand held stylus of GeoMagic Touch. Features extracted from this data are used to train preliminary models, proving that *stroke-hover* information can be extracted from the user's mid-air trajectories. Limitations of classification model developed using this data are addressed through a second data collection study using the Leap Motion controller and a custom hand-held device. Preliminary experiments with latent space intent classification using autoencoders are further discussed. Finally, potential applications of this approach in combination with symbol recognition tasks are discussed.

To identify the *stroke-hover* intention, this research discusses following three approaches (Figure 1.2):

1.2.1 Feature Based Classification

As the first step towards extracting drawing intent from mid-air curves, we first understand user preferences for different postures and drawing mechanisms through a behavioral study using the Leap Motion controller and three interfaces. Findings from the study suggest that users' posture preference varies based on nature of the artifact drawn, and that explicit drawing mechanisms may not be intuitive for the users. Moreover, we identify a geometric structure in the *stroke* and *hover* curves, which forms the basis for development of a feature based *stroke-hover* classification model trained using one-point data recorded from the GeoMagic Touch device (Figure 1.2 (a)).

1.2.2 Raw Data Based Classification

To overcome the limitations of the feature based model, we develop another setup for mid-air drawing data recording using a hand held custom device and Leap controller. The model trained using raw data representations of the tracked palm-wrist-elbow trajectories exhibits higher prediction performance than one point feature based or raw data based models (Figure 1.2 (b)). The prediction results from the 10 dimensional feature model suggest the importance of higher dimensional multi-point data for mid-air drawing intent recognition.

1.2.3 Latent Space Classification

The prediction models trained in the previous two approaches are trained using derived or direct data representations of the recorded data. Using autoencoder-random forest hybrid model, in this third approach (Figure 1.2 (c)), we seek to learn the *stroke-hover* properties from the lower dimensional latent space of the curve data. Prediction results indicate decent classification between the two categories, in turn implying the need for higher dimensional data for better prediction accuracies.

1.3 Thesis Overview

The thesis ahead is distributed as follows. Chapter 2 discusses the current state of art dealing with mid-air curve inputs, and highlights areas where the proposed research is different. Chapter 3 starts with a behavioral study understanding the quantitative and qualitative aspects related to drawing in mid-air. Building upon observations from the study, the chapter further describes a data collection study and intent classification feature-based model. Eliminating the limitations from the feature based model, Chapter 4 describes another data collection study implemented using a custom pen and Leap controller, and subsequent raw data derived models. Chapter 5 quickly introduces the utility of neural networks for drawing intent recognition tasks through latent space classification model, and Chapter 6 finally discusses the broad implications of this work with potential future directions.

2. LITERATURE & PREVIOUS WORK

2.1 Mid-Air 3D Drawing

Sketching, or in a broader sense drawing, is an essential aspect of externalizing or communicating ideas or designs, without the need for a finished product. Initiating from the traditional pen-paper medium, the process of 2D or planar sketching has been effectively replicated using various digital tools. Digital sketching is an extensively studied area of research. In contrast to 3D geometric models, sketches are rough, ambiguous, and vague by nature. Recent advances in augmented and virtual reality, and computer vision techniques have enabled the expansion of sketching as a 2D task to 3D spaces.

A few major modalities govern the creation of 3D sketches. The first category deals with creating sketches using tablet based multi-touch interactions. Bae et al. [18] introduce ILoveSketch, a comprehensive system for expert designers to create and manipulate refined 3D sketches for conceptualization tasks. MentalCanvas [19] allows quick creation of multi-planar curves similar to actual rough sketches. Curve inputs, in general, have also been used as *gestures* by several approaches in multi-touch devices [18, 20, 21, 22, 23]. Lapides et al. [24] discuss the development of a 3-dimensional drawing board metaphor, the *3D tractus*, that allows creation of 3D objects using simple tablet based interactions mounted on top of a board whose height is tracked using sensors. However, these techniques are not always scalable to spatial inputs due to the additional uncertainty added by the third dimension: the intended planarity of strokes is not guaranteed.

With advances in augmented and virtual reality, several techniques have been proposed that use 3D input in virtual CAVE environments for creation of 3D curves [25, 26]. *NapkinSketch* [27] introduced a novel technique for drawing multi-planar shapes in an augmented reality environment, where all major sketching tasks were performed using a tablet based interface. The availability of devices such as HTC Vive and Microsoft HoloLens has enabled development of 3D sketching and modeling applications like TiltBrush [28] for expert designers in an immersive environment.

An initial work that introduced a shift from traditional pen-paper based drawing to more tangible 3D input was proposed by Sachs et al. [29] *3Draw* used six-degree of freedom sensors and custom workstations that allowed designers to input curves mid-air. Kiyokawa et al. [30] introduced the idea of using custom made 3D trackers for manipulation of objects in a shared 3D environment. Recent works by Grosman et al. [31, 32, 33] propose a physical tape drawing metaphor for automotive curve design, and this approach involves using robust hand trackers for hand skeleton detection. 3D input techniques for large displays as described in [34] make use of infrared trackers and Wiimotes for improved robustness of the interactions. Schkolne et al. [35] suggest a 3D drawing system that uses hand motions and tangible tools for sketching and manipulations of 3D curves. Keefe et al. [36] demonstrated a haptics enabled bi-manual interactive system controlled creation of 3D line illustrations. While these devices allow tangible input, they lack the intuitiveness associated with bare hand interactions. The presented research aims at extracting intent from purely bare hand interactions, thus eliminating the possible usage of such devices.

While using bare hand interactions for sketching can be effective for short interactions such as object selection, lack of tangibility makes them difficult for precise tasks like 3D drawing. Finally, unlike sketching on a tablet with a finger or stylus, using a spatial device or a hand posture is not necessarily natural [4]. Works such as Data Miming [16] and grasp-based virtual pottery [17] have demonstrated that for continuous and free-form tasks such as design, users' movements are guided by their interactions with the physical world rather than actions prescribed by the interface designer.

2.2 Symbol & Gesture Recognition

With the increasing availability of interactive devices such as Wii Remote, Microsoft's Kinect, and Leap Motion, there has been rising interest in techniques for mid-air curve input for gesture recognition [9, 10, 11, 12], hand-writing [5, 6, 7], and sketching [2, 4, 3]. Despite the significant advances, hand pose estimation and skeleton tracking still lack the required robustness for simple tasks such as sketching. Further, posture recognition is not scalable for interactions with large displays [37, 38, 39], making it difficult to use multiple body skeleton tracking controllers to identify the user intent. However, as Taranta et al. [9] note, the particular effectiveness of such recognizers for segmentation and recognition of curve inputs on touch surfaces have not been particularly successful in higher dimensional spaces.

Most techniques are developed and evaluated using segmented data, rather than considering a continuous time-stamped data sequence of 3D points [13, 14]. To address this issue, Krishnan et al. [15] develop and evaluate a sliding window based approach to perform activity recognition with streaming sensor data. The approach adopted in JackKnife [9] proposes implementation of dynamic time warping for continuous dynamic identification of gestures. In similar light, Behera [40] et al. introduce a similar approach for signature "spotting" using window based feature sequence analysis. While JackKnife [9] treats each gesture as a time series representation of direction vectors and then classifies with regard to all templates stored in a database, our work processes mid-air curve input as sequential points which have individual set of feature vectors, and performs the classification on each point as and when new points are recorded.

Jacob and Wachs [41] introduce a contextual hand gesture recognition technique for navigating MRIs inside the operating room. Using body posture and hand trajectory movements with gesture spotting networks, their technique helps "spot" or discriminate between intentional and non-intentional cues for navigation. Our approach on the other hand uses pure geometric data of the user's hand motion with no pre-defined postures/gestures to characterize the intention for drawing.

2.3 Object Manipulation & 3D Modeling

Mid-air bare hand gestures have been widely used for expressing user intent [42, 43, 44]. Initial works like *Gesture VR* [42] introduced vision based hand pose extraction techniques for 6-DOF object manipulation in 3D space. Standard computer aided design involve specifying 6 degrees of freedom for 3D objects for standard tasks like assembly. Recent works like *6D hands* [43] use a bi-manual hand tracking system enabling 6-DOF manipulation of 3D assembly components. Vinayak et al. [17, 44] discuss present geometric techniques to extract 3D object manipulation intent based on mid-air bare-hand interactions with the virtual 3D model. The idea behind this research is to build upon observations from these later researches and develop a scalable approach that can potentially be used for mid-air object manipulation and modeling.

2.4 Activity Recognition

Activity recognition is broadly performed using two techniques: vision based, and sensor based. Most vision based techniques use single or multiple cameras for tracking the user's activities. Several techniques involving hidden Markov models (HMM) have been abundantly developed that allow tracking different activities like recognizing ballet steps [45], recognizing tennis stroke [Yamato et al] [46], and Tai' Chi movements [47]. Similar works have been presented to identify typical gestures encountered during human-computer communication [48, 49]. Aggarwal and Xia [50] provide a detailed overview of various vision based 3D activity recognition techniques.

There has been a rising interest in using body worn sensors for gesture or activity spotting from a range of actions that the user performs in a given course of time. Junker et al. [51] present a HMM based two stage classification model which first separates instances where any potential activity is being done, and then classifies those activities using a set of pre-defined templates. Similarly, while Cakmakci et al. [52] tried to identify when a person is looking at their watch, Lukowicz et al. [53] presented a technique for identifying different workshop activities like sawing, drilling, hammering, etc. Lara and Labrador [54] present a comprehensive comparison of 28 systems performing human activity recognition using wearable sensors. Our drawing intent recognition approach is complementary, in the sense that the "anomaly" in the drawing case is when the user has started or stopped drawing, and is performed without using any body worn sensors.

2.5 Complementary Approach

Previous and current works in the domain of gesture, activity, and intent recognition in mid-air tasks follow a set approach. Through any of the above discussed modalities, a pre-defined template of activity/gesture/symbol is recorded and trained. For every newly recorded continuous data stream, fixed length window based features are extracted and matched using the trained template. The results then indicate what shape was being drawn, or what gesture/activity was performed

(Figure 2.1).

In contrast to the traditional approaches, this research treats mid-air curve inputs on a point-by-point basis. Instead of training a template matching algorithm, a binary classifier is trained on appropriate geometric and temporal features characterizing the *stroke-hover* intent from the recorded 3D mid-air data. Then, for every new *point* recorded in a sequence of mid-air data, the trained classifier predicts the drawing intent from extracted features, thus ensuring a seamless point-by-point prediction model for continuously recorded mid-air data.

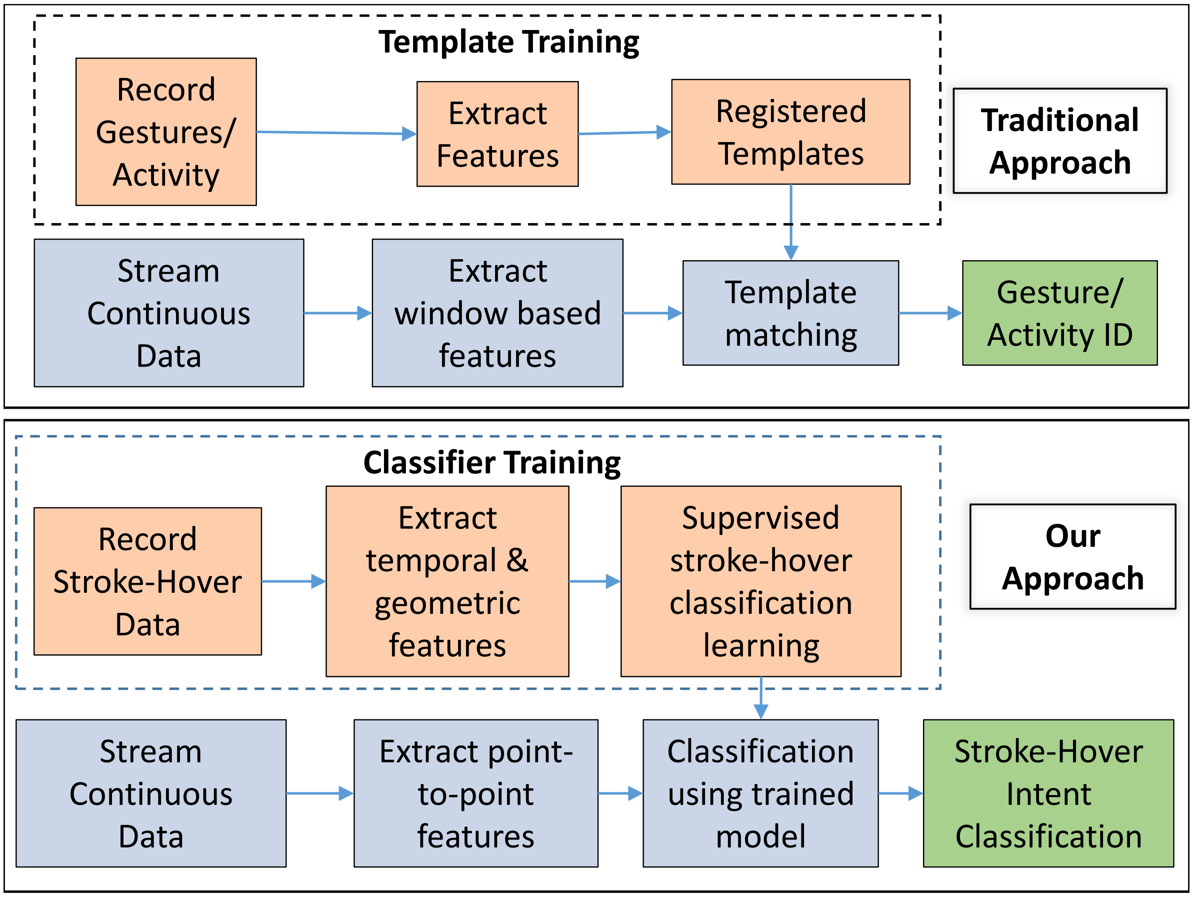

Template Training
Record Gestures/ Activity
Extract Features
Registered Templates
Traditional Approach
Stream Continuous Data
Extract window based features
Template matching
Gesture/ Activity ID
Classifier Training
Record Stroke-Hover Data
Extract temporal & geometric features
Supervised stroke-hover classification learning
Our Approach
Stream Continuous Data
Extract point-to-point features
Classification using trained model
Stroke-Hover Intent Classification

Figure 2.1: Mid-air curve processing methodology followed by traditional template matching and proposed point-to-point classification approaches.

3. FEATURE BASED STROKE-HOVER CLASSIFICATION

To address the problem of identifying the user's drawing intention from their hand trajectories, it is first necessary to understand how users draw in mid-air. In this chapter, we first describe a behavioral study discussing user preferences towards gestures and defined mechanisms for drawing in mid-air. Based on the findings from this study, we explore different characteristics of the recorded mid-air curve to extract the *stroke-hover* intention, and describe a feature based model to achieve the same. Finally, we look at some prediction results using this feature based model, and discuss limitations with this approach [1].

3.1 Mid-air Sketching: Observational Study

To better motivate the need for the proposed approach, an observational study is conducted with an intentions to observe how users specifically react to known spatial input conditions (such as constraining the sketch on a canvas or using a specific gesture) in comparison to a completely rule-free scenario (how one might describe an object through spatial movement without a computer interface at all). To achieve this, three interfaces using the Leap Motion Controller are implemented to record the trajectory of the user's hand skeleton while drawing a given curve. Based on the recorded hand position, the following rules were applied for detecting whether the recorded point is stroke or hover:

1. *Proximal Plane* (**I1**): A trajectory point is considered as a stroke point if the palm is in proximity to a pre-defined sketching plane within a defined threshold (Figure 3.1 (a)). The interface recognizes the user's intention to sketch when the palm is within a certain threshold of the front facing plane on the screen. Even though this method is agnostic to any specific pose of the hand, users were asked to assume a pointing posture while sketching to maintain

[1] Part of the data reported in this chapter has been reprinted with permission from Association for Computing Machinery, Inc. and has been extracted from "To Draw or Not to Draw: Recognizing Stroke-Hover Intent in Non-instrumented Gesture-free Mid-Air Sketching" by Umema Bohari, Ting-Ju Chen, and Vinayak, In 23rd International Conference on Intelligent User Interfaces (IUI '18). ACM, New York, NY, USA, 177-188. DOI: https://doi.org/10.1145/3172944.3172985

a natural way of providing input.

2. *Pinch Posture* (**I2**): A trajectory point is considered as a stroke point if the hand assumes the pinching posture (Figure 3.1 (b)). The choice of the pinch posture is motivated from the way one holds a pen while sketching on a piece of paper. The pinch is recognized when the Euclidean distance between the thumb and index finger is within a pre-defined threshold.

3. *Unrestricted* (**I3**): In the third interface, the users were simply asked to describe a curve in the air without any restrictions on their hand movement or posture. Here, there was no explicit distinction between hover and stroke for the recorded data (i.e. all points were stroke points).

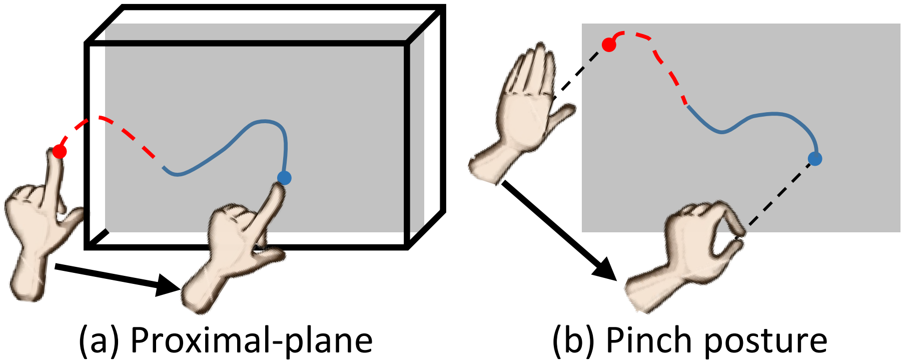


Figure 3.1: Illustration of the plane-proximity and pinch gesture mechanisms used in Leap Motion interfaces I1 and I2 for the observational study.

3.1.1 Participants

10 engineering students (5 female) within the age range of 19-30 years were recruited. Except one participant, none of these participants had prior experiences with motion tracking devices such as Wii and Kinect or mid-air sketching.

3.1.2 Procedure

The total time taken during the experiment varied between 30 and 35 minutes and the three interfaces were randomized across the participants. After describing the setup, and the purpose of the study, the features of the first sketching interface were explained to the participants, and its usage was demonstrated. For each participant and task **T3**, a video of the task, the completion time, and the time-stamped 3D coordinates of the trajectories generated by the users for each sketched shape were recorded. Each participant performed the following tasks:

1. *Unrestricted Sketch* (**T1**): Participants were asked to draw primitive shapes, like a square, in mid-air without restrictions on their hand posture or movement. The participants' responses were video recorded for observational exploration.

2. *Posture-preference* (**T2**): Participants were then asked to repeat the primitive sketching, but with one or more hand postures of their choice from a list of pointing, two-finger pinch, open palm, and pen-holding posture. In addition to video recording, the reasons for these choices were also recorded.

3. *Practice* (**P**): To familiarize themselves with the interaction of their hand postures with the corresponding three interfaces, the participants were given a brief demonstration of the software and its functions, and were allowed to practice for 5 minutes on each interface. They were allowed to ask questions and were provided guidance when required.

4. *Sketching with **I1** & **I2*** (**T3**): Participants were asked to sketch on the planar-proximity and pinch based interfaces in a randomized manner. For each of these interfaces, every participant sketched at least two primitives (Figure 3.2). Although the duration of time for each interface was set to five minutes, the participants were allowed to sketch more shapes with their aspirations. The canvas was cleared after completion of each primitive.

5. *Questionnaire* (**Q**): Finally, each participant answered a series of questions regarding their perception of each of the interfaces in terms of ease of use, intuitiveness, and robustness.

Open-ended comments regarding the tasks were also recorded.

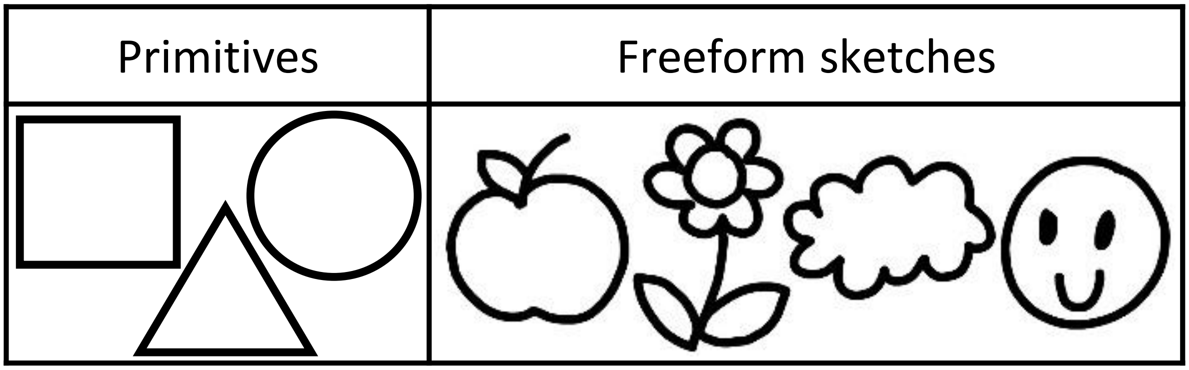


Figure 3.2: Primitives and free-form multi-stroke sketches drawn by users during the observational study.

3.1.3 Findings

With each participant drawing 6 sketches, a total of 106670 points (55919 *stroke*, 60751 *hover*) were recorded. We make the following observations:

1. *Posture Comparison*: As expected, nine out of ten participants used index finger to draw single-stroke primitives in the unrestricted sketch interface (**I3**). However, in the posture-preference task (**T2**), three out of these nine participants who used the pointing posture in **I3**, changed their preference to the pinching posture. One user stated: *"It is more like using pen and paper"*. Another user stated: *"Pinching can help me deal with more complex, detailed drawings"*. This indicates that (a) there is a natural *default* hand posture and body movement that manifests commonly across users during mid-air curve input and (b) the way users move while sketching in air is dependent on the nature of the artifact itself that they are trying to draw.

2. *Interface Comparison*: We further observed that the order of interfaces affected ease of adopting different hand postures in mid-air sketching. A participant who completed sketching tasks with **I1** and **I2** before **I3**, stated: *"It is difficult to turn it on and off, people normally*

do not change their hand posture despite it is a stroke or a hover". Nine participants out of ten also preferred **I3** over the other two even though I3 did not have any visual cue that distinguished hover from stroke. They stated: *"The normal interface was easiest"* and *"I do not need to worry about gestures and everything was detected with the normal interface"*.

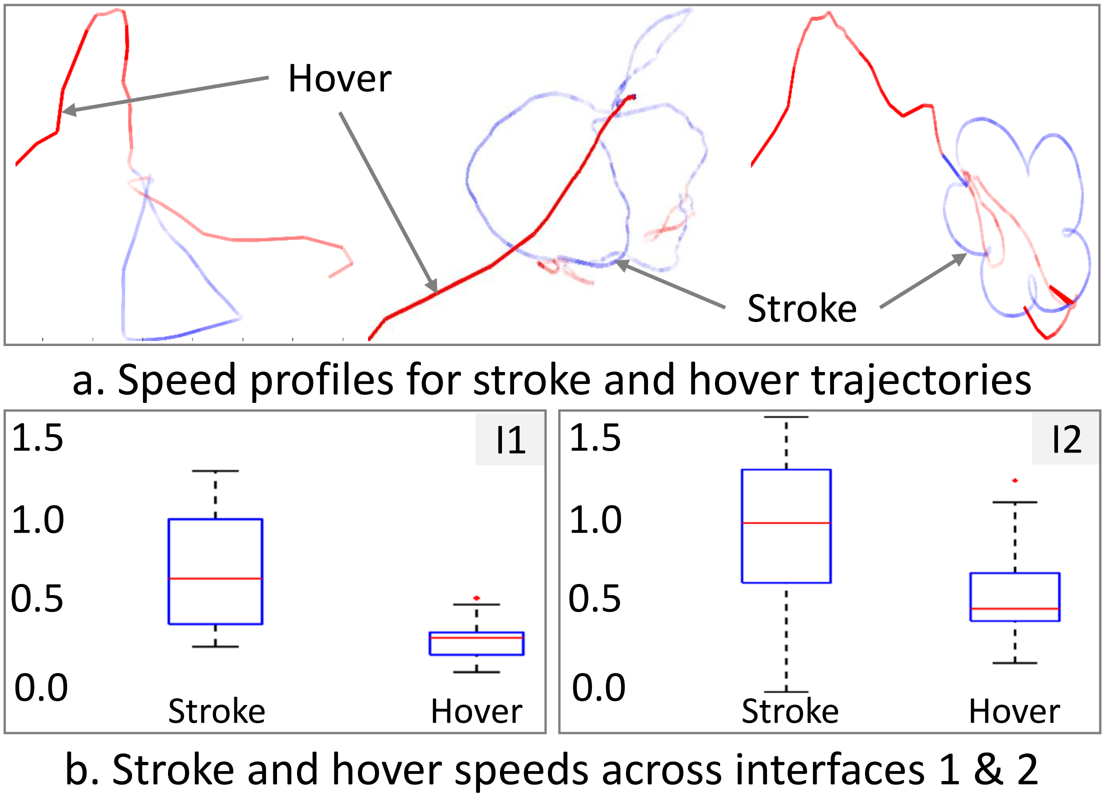


Figure 3.3: The speed profile (top) shows a near-constant speed for hover that is greater than stroke speeds. The average hover and stroke speeds were greater for the pinch-posture interface I2 in comparison to proximal plane (I1)

3. *Motion Characteristics*: As expected, participants were generally slower (Figure 4.3(a)) while creating strokes (**I1:** 0.35 m/s, **I2:** 0.56 m/s on average) as opposed to hover (**I1:** 0.66 m/s, **I2:** 0.9 m/s on average). For the proximal plane interface (**I1**), this was observed to a larger extent in comparison to pinch interface (**I2**) (Figure 4.3(b)). Further, the uncertainty of users' trajectory increased as they reached closer to the instance of transitioning

from hover to stroke. This was observed in terms of large straight hover trajectories followed by short zigzag ones while transitioning from hover to stroke.

4. *Shape Type*: There were significant differences in how participants approached different types of primitives. They spent time in refining details for multi-stroke shapes. The distribution of hover and stroke regions trajectories for drawing single-stroke primitives were common across users when compared with general multi-stroke shapes. This strongly indicated that for arbitrary sketches, there is a need for a general computational strategy for segmenting and recognizing meaningful parts of the users' trajectory.

3.2 Stroke-Hover Modeling Approach

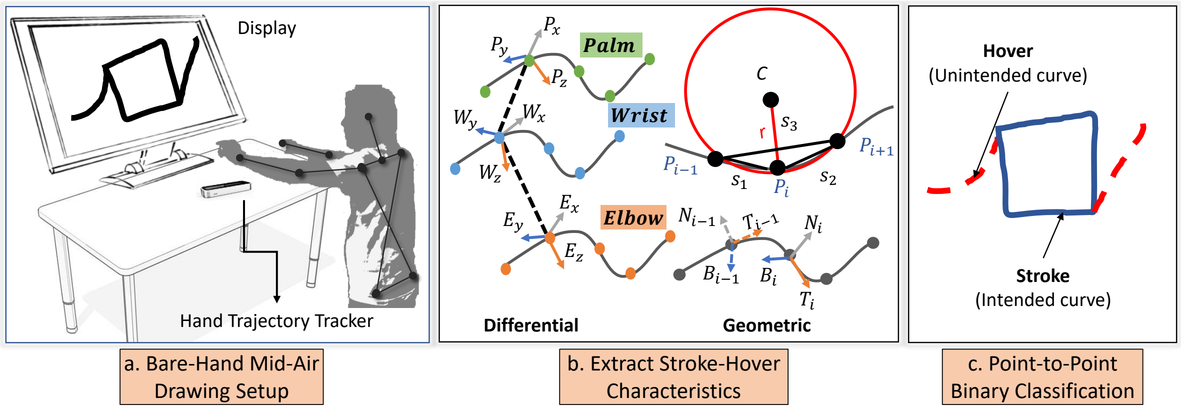
Findings from the behavioral study indicate the necessity of developing an approach for mid-air curve input that minimizes the use of pre-defined hand postures/gestures, or specialized interaction design to initiate or stop drawing in mid air. Speed profiles and average completion times of the hand trajectory data recorded points towards the different temporal characteristics of *stroke* and *hover* curves. The problem thus now simplifies to developing a model that extracts important geometric and temporal features from the recorded data, and identifies the drawing intent for every recorded point. The *stroke-hover* identification work flow (Figure 3.4) can be explained as follows:

1. *Record Mid-Air Drawing Data*: The mid-air drawing data is recorded by using a depth or motion tracking camera like the Kinect or Leap Motion controller.

2. *Extract Features for Model Training*: From the recorded 3D data, distinguishing geometric and temporal properties of *strokes* and *hovers* are computed. Different combinations of these features are used to train binary classification models.

3. *Point-to-Point Classification*: For every new point recorded, the trained classifier is now used for classification.

4. *Report Drawing Intent*: Finally, the stroke-hover intent for every classified point is reported.



a. Bare-Hand Mid-Air Drawing Setup

b. Extract Stroke-Hover Characteristics

c. Point-to-Point Binary Classification

Figure 3.4: Stroke-Hover Intent Recognition Workflow.

Once classified, the *strokes* can be used for further processing like shape retrieval, recognition, and search tasks, to name a few.

3.3 Mid-Air Drawing Data Collection Study

The lack of robust finger tracking was a major concern for the pinch-posture interface (**I2**). As a result, stroke points were intermittently lost due to incorrect prediction of the pinch posture. One of the participants, who had experience in motion tracking devices such as Wii and VR, stated: *"The software interface with hardware probably had some minor issues with detection of fingers...It couldn't follow the movement of my hand and veered off course many times"*. In contrast to the behavioral study, we used the GeoMagic Touch device for data acquisition. Here, users specified their intent to sketch through the use of buttons on the device stylus (similar to a mouse click-and-drag). This choice of hardware was directly a result of the lack of robustness observed in our behavioral studies with the Leap Motion wherein capturing user intent was prohibitively difficult.

3.3.1 Participants

We recruited 21 engineering students (11 female) within the age range of 19-30 years. None of these participants had prior experience with using the GeoMagic Touch device.

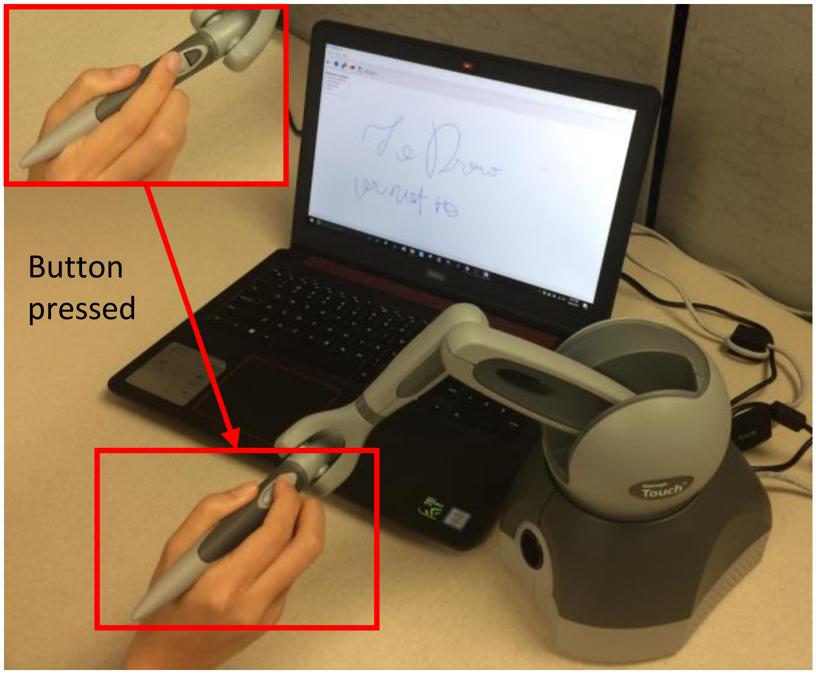


Figure 3.5: 3D Sketching data collection setup using GeoMagic Touch device. Inset: Button press operation used for distinguishing hover from stroke.

3.3.2 Sketching tasks and assumptions

Each participant was asked to draw 47 different shapes comprising of symbols, 2D and 3D primitives, and free-form sketches in mid-air (Figure 3.6). The participants were instructed to sketch the curves as naturally as they could (i.e. as fast or as slow as they would if there were no interface). We assume that the 2D data sketched by users is primarily planar, and can be drawn using single strokes or multi-strokes; while the 3-dimensional primitives recorded are multi-planar and multi-stroke sketches.

3.3.3 Recorded Data

We developed a simple interface that allows the participants to sketch in 3D using the GeoMagic Touch stylus (Figure 3.5). For every curve drawn, the interface records a continuous sequence of 3D coordinates of the stylus tip trajectory $P_{x,y,z}$, and the classification of that point as being *stroke*(1), or *hover*(0). While writing on a piece of paper, or sketching on tablet surfaces,

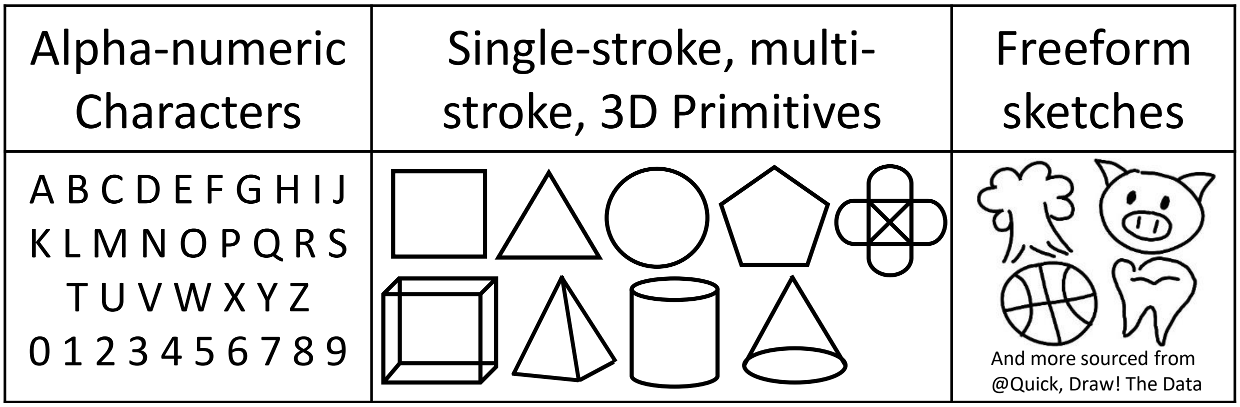


Figure 3.6: Symbols (left), 2D and 3D primitives (center), and free-form 2D sketches (right) drawn by users during 3D sketching data collection using the GeoMagic Touch device.

velocity of the traversal trajectories between two successive strokes is much faster than the stroke trajectories themselves. As noted by Johnson et al.[55], time data is generally used for segmenting strokes in different sketch recognition algorithms. Speed profiles of the data recorded in the observational studies (Figure 4.3), also point towards a similar distinguishing factor between *stroke* and *hover* curves. Therefore, along with the curve coordinates and classification status, we record the time stamp in milliseconds t of all drawn points.

3.4 Feature Design & Model Training

In machine learning, a *feature* can be defined as an individual measurable property or characteristic of a particular pattern being observed. Based on the motion profiles and local geometric properties of the recorded 3D temporal data, to train and test the *stroke-hover* classification model, a feature list comprising of the following curve characteristics is constructed:

1. *Motion profile*: For every given point *i* recorded on the trajectory, its speed (s_i), acceleration (a_i), and jerk (j_i) relative to the $(i-1)^{th}$ point is computed:

$$s_i = \|P_i - P_{i-1}\|, a_i = \frac{s_i - s_{i-1}}{t_i - t_{i-1}}, j_i = \frac{a_i - a_{i-1}}{t_i - t_{i-1}}$$

Angular velocity and three axial acceleration are found to be good measures for capturing human activity [56].

Based on the motion characteristics of the 3D data recorded in the observational study, it is noted that at the *stroke-hover* transition, there is an abrupt shift in the local curve speed. To capture this shift, the relative speed ratio (S_r) at a given point is calculated as:

$$S_r = \frac{s_i}{s_{i-1}}$$

2. *Curvature*: Recorded curve trajectories from the Leap Motion observational study, and GeoMagic Touch data collection study are suggestive of the fact that *hover* trajectories have a higher degree of flatness, whereas *stroke* trajectories have a higher curvature. This measure of flatness of the curve at the i^{th} point is captured by estimating the Menger curvature (c_i) at that point (Figure 3.7 (a)). For a given triangle formed by P_i, P_{i-1}, P_{i+1}, the curvature is given by:

$$c_i = \frac{1}{r} = \frac{4A}{s_1 s_2 s_3}$$

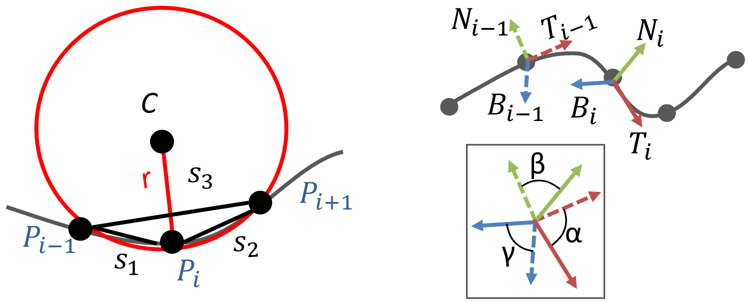
where *A* is the area of $\triangle\ P_{i-1}P_iP_{i+1}$, and s_1, s_2, and s_3 are lengths of the triangle's sides.

3. *Rate of change of Frenet frame*: Three orthogonal components (tangent t_i, normal n_i, binormal b_i) of the Frenet frame (Figure 3.7 (b)) associated with each edge forming the curve are further computed as:

$$t_i = \frac{P_i - P_{i-1}}{\|P_i - P_{i-1}\|}, b_i = t_{i-1} \times t_i, n_i = b_i \times t_i$$

As the user draws the *stroke* and *hover* curves, the change in planarity is estimated by calculating the rate of change of angles between the consecutive Frenet frame components. These angular velocities (ω_α, ω_β, ω_γ) form the last three components of the training and testing feature vector:

$$\omega_\alpha = \frac{\alpha_i - \alpha_{i-1}}{t_i - t_{i-1}}, \omega_\beta = \frac{\beta_i - \beta_{i-1}}{t_i - t_{i-1}}, \omega_\gamma = \frac{\gamma_i - \gamma_{i-1}}{t_i - t_{i-1}}$$



a. Curvature Estimate b. Frenet frame rotation

Figure 3.7: Estimated curvature and discrete Frenet frames of the recorded curve.

3.5 Training and Testing

Of the total data recorded from 21 participants, data from 2 participants had to be discarded, due to misrepresentation of visual cues that hinted towards it being a plane drawing task instead of mid-air sketching task. Of the remaining data, 50000 points (22923 *stroke*, 27078 *hover*) were used for training the model, while the remaining 50000 points (24292 *stroke*, 25708 *hover*) were used to test and cross-validate the data. While training, the feature vectors were randomly sampled from the available pool to eliminate any kind of over-fitting of the trained model due to adjacent data points. For testing, sequences of points belonging to a given shape were sampled. All machine learning classifiers are implemented using MATLAB's Statistics and Machine Learning Toolbox [57] and implementations. With mid-air sketches essentially being a series of alternating *stroke* and *hover* curves, it is assumed that the status of a given point on the curve is dependent on that of its neighbors. To test this assumption, a *k*-Nearest Neighbor binary classifier [58] is trained. It is ensured that there is an equal mix of *stroke* and *hover* features in the training list to eliminate model bias towards the over-represented class. Since Support Vector Machines (SVM) [59] find utility in multiple gesture recognition tasks, a binary SVM classifier is also trained. Training and testing data is uniformly scaled for accurate implementation of SVM. Finally, random forest classifiers [1] are explored, and the effect of different tuning parameters for accurate predictions is discussed.

3.5.1 k-NN Classifier

In pattern recognition, k-NN or k-Nearest Neighbor is a non-parametric method used for classification or regression tasks [60, 61]. For data distributed in a given feature space, the k-NN classifier outputs the classification of a given point. The classifier locally computes the distance of the point under consideration from its neighbors, and classifies the point to the category that is most common among its k closest neighbors. To enhance the predictions from this instance-based learning method, the distance computation of a point from its nearest neighbors is weighted, to ensure that the contributions of the closest points is enhanced. This makes the algorithm sensitive to the local structure of data - closely controlled by the parameter k - and the trained classifiers reflect this structure during their predictions.

3.5.2 Support Vector Machines

Support Vector Machines are supervised learning algorithms that are typically used for binary classification tasks [62]. For a given data distributed in a *linear* feature space, SVMs try to find a linear separating boundary that divides the data into two parts belonging to each class. If the data distribution is non-linear, SVMs use techniques like kernel tricks and implicit mapping to first transform the data into linear high-dimensional space, to identify the best separating boundary. For every newly added point in the dataset, the classifier calculates its distance from the separating boundary or hyperplane, and assigns it a classification accordingly. In other words, the SVM constructs a hyperplane, or sets of hyperplanes, as far as possible from the closest points in each of the two classes, to achieve good generalization of the data.

3.5.3 Random Forests Classifier

Random forests belong to the category of ensemble learning methods used for classification and regression tasks [63]. For classification tasks, random forests use bootstrapped samples to train decision trees, and classify every given feature based on the maximum class vote received from the trees collectively. A decision tree is a flowchart like structure that assigns a class label for every observation by repeatedly splitting the data from root to final leaf node based on certain classification rules. A critical feature of random forests is that they are invariant to feature data scaling, can handle transformations of the feature space, and are robust to inclusion of irrelevant features [61]. The low bias, high variance nature of decision trees is effectively compensated for by bootstrap aggregation, or bagging, used in random forests, and deep grown trees have known to identify good data generalization during supervised learning tasks.

3.6 Stroke-Hover Classification Results

In this section, the prediction accuracies of the 3 classifiers are discussed, and the best model is selected. Further, a hyper-parameter search is performed to identify the most optimum parameters of the random forest model. Finally, *stroke-hover* predictions using this model are described and limitations of the current model are discussed.

3.6.1 Classifier Comparison

To compare the classification accuracies (η) across the 3 models, symbols, 2D and 3D primitives, and free-form sketches from the testing feature list are sampled and tested.

After performing parametric optimization, with 27 closest neighbors, and *correlation* distance metric, the *k*-NN model predicts with an average accuracy of $\eta = 57.11\%$ at a 0.04 seconds per point prediction rate. For cross-validated two-class SVM models using both radial basis function, and Gaussian kernels, the average prediction accuracy is $\eta = 51.44\%$ at a 0.02 seconds per point prediction rate. It is observed that while the *k*-NN model shows some trends of segmenting the stroke-hover data (Figure 3.8 (1a),(2a),(3a),(4a)), the SVM model classifies every point as being *stroke* (high false positive) (Figure 3.8 (2b), (3b),(4b)). The random forest model, on the other hand, exhibits good demarcation between *stroke* and *hover* curves, and classifies them with an average accuracy of $\eta = 71.63\%$ at 0.03 seconds per point prediction rate.

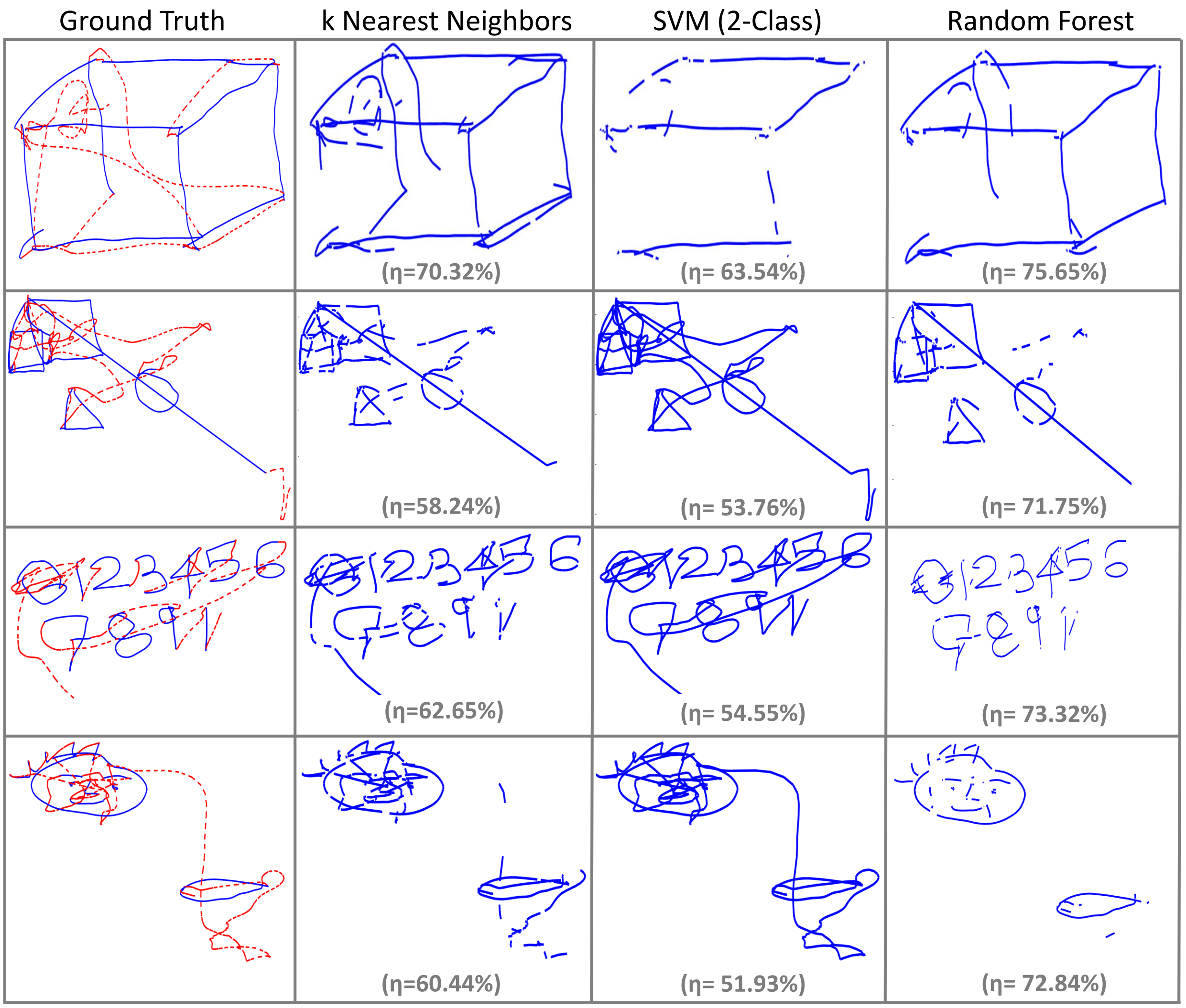


Figure 3.8: Sampled prediction results with accuracies (η) for k-NN, SVM, and Random Forest binary classifiers. k-NN and SVM models show high false positives.

3.6.2 Random Forest

The random forest classifier is explored to further improve its accuracy, by training different models using the following feature list:

$$F_i = \begin{bmatrix} s_i & a_i & j_i & c_i & S_r & \omega_\alpha & \omega_\beta & \omega_\gamma \end{bmatrix} \tag{3.1}$$

Two main tuning parameters – maximum node splits per decision tree N_s, and number of trees

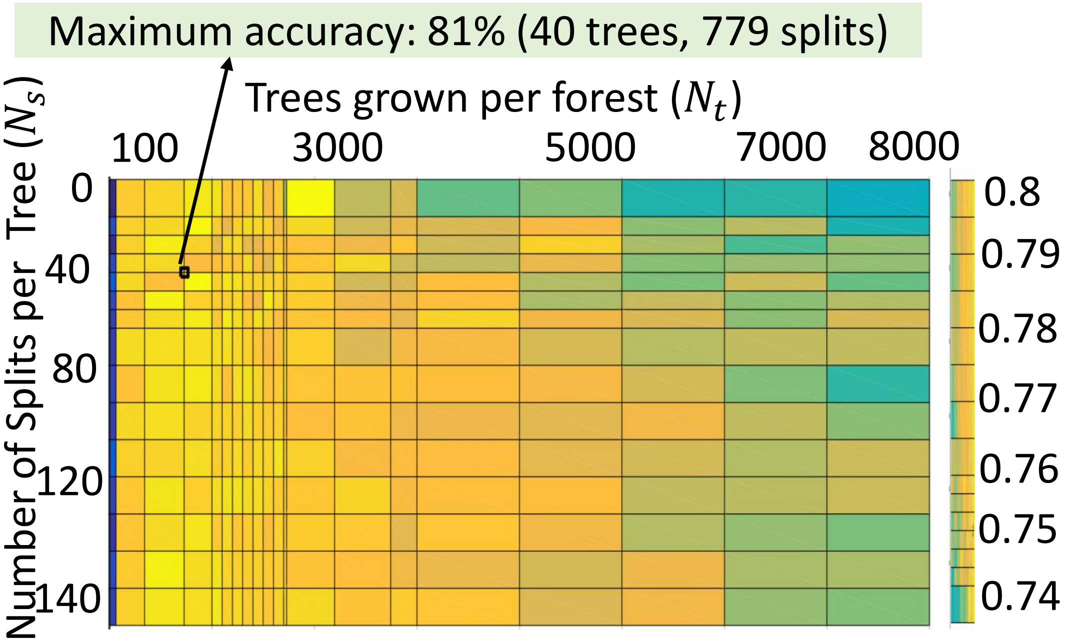


Figure 3.9: Variation in random forest model accuracy (η) with changes in number of individual tree splits (N_s) and number of trees per forest (N_t).

per forest, N_t, are used to control the prediction accuracy. The depth of every decision tree is controlled by N_s. Starting with the default values of $N_s = 7$ and $N_t = 10$, a grid search is performed to identify optimum parameters. Based on this analysis, the model with $N_s = 779$ and $N_t = 40$ performs best with a prediction accuracy of $\eta = 82\%$ for 50000 points (Figure 3.9).

As in the case of SVM, the effect of scaling 3D data coordinates as well as processed features, is explored on the model prediction accuracy. It is observed that the model with no scaling performs with the best accuracy. Speed (s_i), Speed ratio (S_r), local curvature (c_i), and ω_α are found to be the dominating features, and the final random forest model is retrained using only these parameters.

3.6.3 Instrumented Controller Predictions

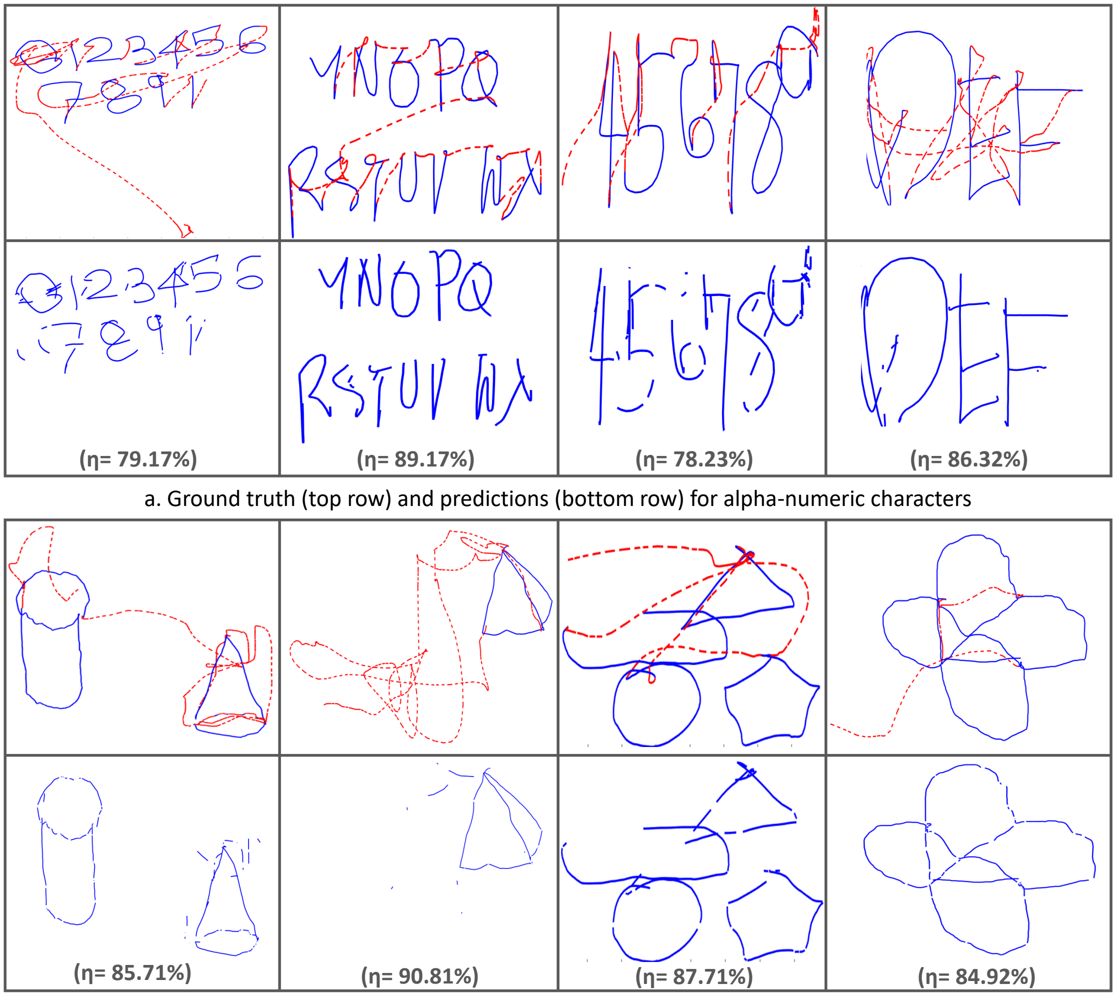
The optimum random forest model is tested using symbols, 2D and 3D primitives, and free-form shapes from the testing feature list. Higher prediction accuracy is observed in case of primitives and free-form shapes $\eta = 85.75\%$, as compared to symbols $\eta = 78.75\%$. This can be explained in a couple of ways. First, during the data collection study, it was observed that due to familiarity with symbols, participants sketched them at a much faster rate. On the other hand, the participants were found to be cautious while sketching multi-stroke free-form sketches and 3D primitives. This factor had a direct effect on the hypothesis that *hover* trajectories should essentially be traversed faster as compared to *stroke*, in the sense that, this was followed for the multi-stroke shapes, but was not reflected for the recorded symbols data. Second, from a close analysis of the data collection video, it was observed that symbol sketches were primarily restricted to a single plane, while multi-stroke primitives and free-form sketches spanned multiple planes. While the multi-planarity was effectively captured for the shapes using the three Frenet frame vectors, the same cannot be guaranteed for symbols.

3.6.4 Bare Hand Predictions

Performance of the model is further evaluated by predicting 3D sketching data recorded in the Unrestricted Interface **I3** using the Leap Motion controller. The predicted results have two characteristics: discontinuity in predictions, and high false positive rates (hover misclassification). These characteristics can be attributed to the different ways in which the training and testing data sets are recorded on the GeoMagic Touch and Leap motion interfaces, respectively. On one hand, while the GeoMagic Touch based interface recorded data with accurate sketch-hover demarcation, it ultimately is a tethered device, with limited work volume availability. On the other hand, with the Leap interface, users had comparatively more freedom with respect to workspace availability, and were able to draw curves at a much faster rate. However, it is worth noting that the model was able to classify sufficient points without geometrically pre-processing the recorded data using techniques like corner detection.

3.7 Conclusion

The primary reason for using the haptics device for 3D sketching data collection was to ensure accurately recording the *stroke-hover* data of the drawn points. However, the device itself posed certain limitations due to its physical manipulation capacities, offering roughly an interaction space of 16 in x 12 in x 10 in. Also, since it was primarily tethered to the computer, the user did not have



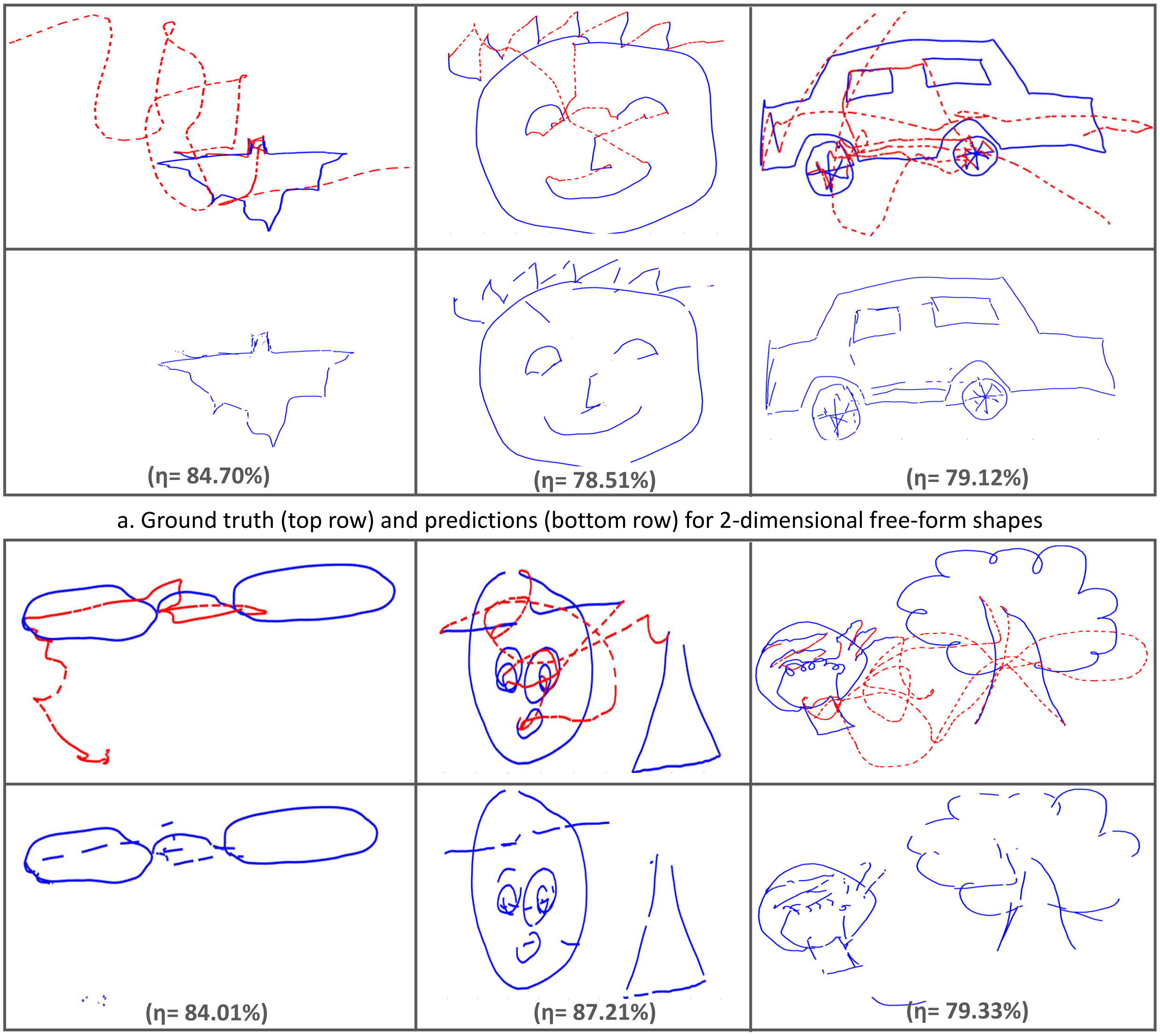
a. Ground truth (top row) and predictions (bottom row) for alpha-numeric characters

b. Ground truth (top row) and predictions (bottom row) for 3D and 2D single- and multi-stroke shapes

Figure 3.10: Random Forest model prediction accuracies (h) for symbols (first two rows) and 3D/2D primitives (bottom two rows) sketch data recorded using GeoMagic Touch.

much freedom in terms of moving away from the computer and sketching. It was also observed that participants experienced high resistance from the stylus pivot while traversing wavy curves like the tree crown (Figure 3.11). This resulted in the user dwelling for much longer time while drawing *stroke* curves. Restricted motion about the stylus pivot also resulted in wavy hover trajectories, resulting in high curvature *hover* curves. Further, mid-air curve input speed is dependent on some other factors like the type of curves, preciseness with which they are drawn, and the application for the 3D curve. These factors are taken into consideration in the study discussed in the next section.

Based on the results obtained from the offline instrumented controller and Leap bare hand data



a. Ground truth (top row) and predictions (bottom row) for 2-dimensional free-form shapes

b. Ground truth (top row) and predictions (bottom row) for 2-dimensional free-form shapes

Figure 3.11: Random Forest model prediction accuracies (η) for multi-stroke free-form sketch data recorded using GeoMagic Touch.

predictions, modifications in the feature design, as well as the data collection set up are proposed. Moreover, the model in this approach is simply trained using data recorded from a single stylus point moving in 3D space. This low dimensionality of mid-air drawing data may not be sufficient to capture the *stroke-hover* characteristics. To eliminate spatial restrictions caused during data collection task, the GeoMagic Touch with its tethered stylus is replaced by a wireless, Bluetooth connected device, used as a 3D pen. In this setup, the hand trajectory of the user is recorded using the Leap Motion sensor, and the *stroke-hover* intention is recognized through explicit button press operations on the 3D pen. Further, to improve the quality of data captured, we experiment with 3-point tracking of the user's hand.

4. RAW DATA BASED STROKE-HOVER CLASSIFICATION

Results from the model trained using a single point trajectory indicate that *stroke-hover* classification of a given 3D trajectory can be achieved using local geometric features extracted from the drawn curve. This model was trained from data recorded using the tethered stylus of GeoMagic Touch device. However, validation of this model using bare hand data recorded in interface **I3** suggests that using non-tethered devices for recording data closely emulates the actual application scenario. Several works dealing with gesture recognition, and activity recognition suggest using high dimensional geometric data for tasks involving mid-air curve input. Moreover, the activity of drawing in air involves significant movements of the elbow, wrist, and palm trajectories. Based on these observations, this chapter describes a new data collection study implemented using a custom made, hand-held, non-tethered device. Apart from recording the palm trajectory, the user's wrist and elbow trajectories are also recorded. Qualitative and quantitative differences between the two types of data are discussed, and multiple feature construction models for training a *stroke-hover* classifier are described. Finally, the design and selection of feature space, model training and testing, and results from the proposed three point local differential model are discussed.

4.1 Data Collection

4.1.1 Experimental Setup

Based on the feedback received from the preliminary data collection study, the new experimental setup is designed to eliminate the spatial and physical restrictions associated with a tethered device. The setup (Figure 4.1) primarily comprises of the following major components:

1. Hand Trajectory Tracking: In this experiment, instead of tracking a single point in 3D space, three points on the user's hand trajectory are recorded. Bare hand interactions in the mid-air involve a series of simultaneous movements of the user's elbow, wrist, and consecutively, palm joints. To incorporate the effect of these movements in the *stroke-hover* intent classifier, the Leap Motion controller is used to record time stamped coordinates of the user's elbow-

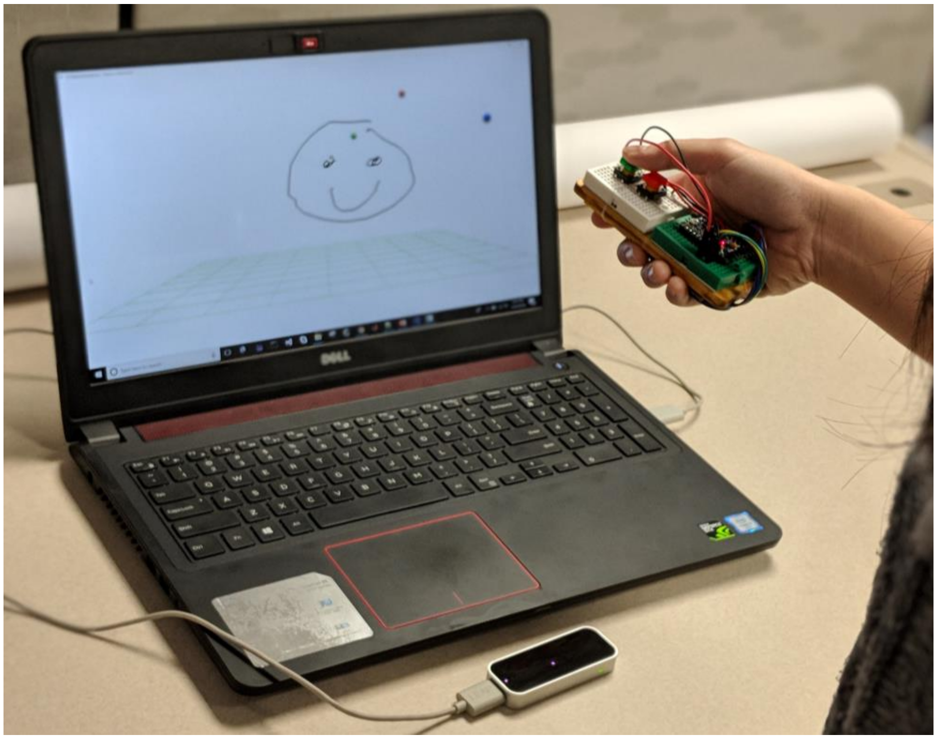


Figure 4.1: Data collection setup using the Bluetooth-connected 3D pen and Leap Motion controller.

wrist-palm inclusive three point trajectory. The Leap controller is placed on the table, and the user draws 3D sketches within the interaction volume of the device (Figure 4.1).

2. Intent Detection: To record the drawing *stroke-hover* intent, a device resembling a 3D handheld pen is developed. The device consists of a button, that is pressed by the user every time a *stroke* is to be drawn. The pen communicates with the computer via a Bluetooth serial port, and is programmed using an Arduino Pro Mini. At any given point of time, the pen sends a 0-1 status to the computer, indicating the button up or down states, respectively. This information, combined with the 3-point trajectory recorded by the Leap Motion effectively constitute a single data point at any given time.

The data recording interface is developed using C++ on the OpenGL platform, and implemented on a Dell Precision Tower 3620.

4.1.2 Participants

For recording 3-point trajectory 3D sketching data, 25 engineering students (13 female) within the age range of 19-30 years were recruited. 3 participants had prior experience with 3D sketching (digital), while none of the participants had experience with any computer vision devices like the Leap motion controller, or 3D depth cameras. None of the participants had any experience with using hand-held devices for drawing in mid-air.

4.1.3 Sketching tasks and assumptions

The tasks for this study were elaborately designed to incorporate multiple geometries, different planarity, and a broad variety of motion and gestural drawings. Each participant recorded data across six shape categories (Figure 4.2) as follows:

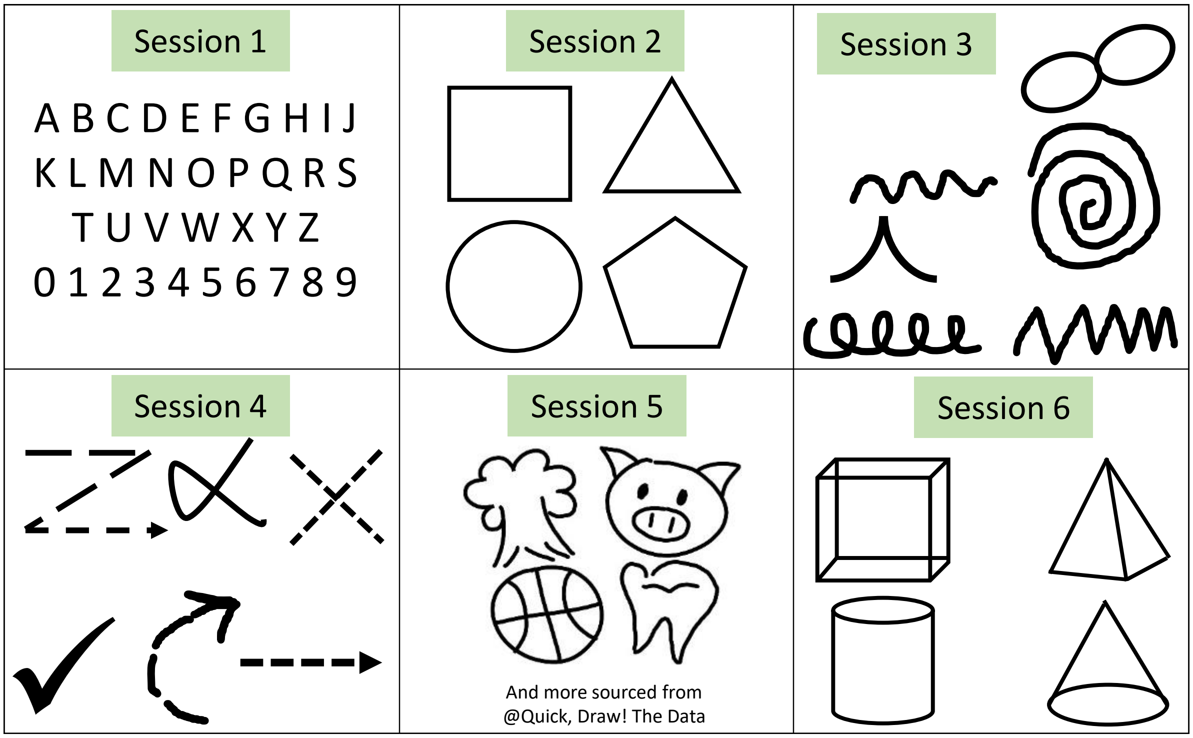


Figure 4.2: Alphanumerics (a), 2D primitives (b), motion gestures (c), special curves (d), 2D free-form shapes (e), and 3D primitives (f) drawn by users during the data collection study.

1. Alphanumerics: In this session, participants were instructed to draw alphanumeric characters.

2. 2D Primitives: This session involved drawing basic 2D primitives. To understand whether the size of a drawn shape affects the *stroke-hover* classification accuracy of the models, the participants were asked to draw the primitives in three sizes: small, medium, and large. All shapes were drawn on the front, top, and right planes to ensure the shape's planarity is incorporated while training the classifier.

3. Gestures: This session involved participants drawing curves with special geometric properties (degree 2, degree polynomials) and features.

4. Special Curves: In this session, the participants used the 3D pen to "draw" motion gestures in mid air. These set of curves were recorded to validate the utility of *stroke-hover* classification approach towards symbol/gesture recognition tasks.

5. Free-Form Data: This session allowed the participants to use the 3D pen to draw free form planar shapes on the front plane, from Google's Quick, Draw[64] database.

6. 3D Primitives: In comparison to the other sessions, here, the participants were not restricted to a given plane, and were allowed to draw 3D primitives within the entire interaction volume of the Leap controller.

Unlike the previous study, no visual feedback about the mid-air curve input was provided to the participants, and instead, they were instructed to draw the curves as naturally as they could (i.e. as fast or as slow as they would if there were no interface). The hand trajectory tracking was monitored by the study proctor, and the participants were instructed accordingly. After every session, the participants were allowed a break to ensure that hand fatigue does not bias the recorded data. It is assumed that the 2D data sketched by users is primarily planar, and can be drawn using single strokes or multi-strokes; while the 3-dimensional primitives recorded are multi-planar and multi-stroke sketches.

4.1.4 Recorded Data

For every curve drawn, the interface records a continuous sequence of 3D coordinates of the user's elbow $E_{x,y,z}$, wrist $W_{x,y,z}$, and the palm trajectory $P_{x,y,z}$, and the classification of that point as being *stroke*(1), or *hover*(0). Along with the curve coordinates and classification status, the time stamp of all drawn points is recorded in milliseconds *t*.

4.1.5 User Feedback & Observations

Once all tasks were completed, the participants recorded their experiences with drawing the designed tasks in air through a questionnaire. Some observations from the questionnaire are as discussed below:

1. *Utility of Mid-Air Curve Input*: The users were asked to describe if they typically used mid-air curve input while describing ideas during a conversation, and their usage frequency. All participants except 3 indicated using hand movements during a conversation. Most participants indicated using hand movements to describe shapes, sizes, or special features of a given idea/product. One participant stated,*"...(I) use them in almost every conversation. I attach shape to concepts. It may change according to the severity of the design."* Another mentioned, *"...have used hand movements to describe assemblies in vehicles during my Senior year project, I use them regularly"*. A Civil Engineering student expressed, *"I recall using hand movements to describe many technical aspects and drawings to my colleagues in Civil Engineering"*. These observations stress upon the utility of mid-air curve input in sophisticated design tasks.

2. *Applications in Design*: When asked about the utility of mid-air curve input in design tasks, most participants pointed towards using them for 3D modeling and concept drawing tasks. One user mentioned:*"...similar to some 3-D based note pad for jotting down all thoughts including complex designs"*. Another user stressed upon the corrective actions that need to be undertaken to make such mid-air input usable for modeling tasks: *"...the application would be good for 3D drawings in the future. For example Solidworks, but the software would need*

to be able to recognize what shape the user is trying to draw and correct it.", while another stated that it can be used for machine design tasks, if there is a way to recognize the shape that is being drawn. One participant stressed upon the importance of such techniques for engineering specific tasks: *"There are endless possibilities. Particularly as a Mechanical Engineer I would say solid modeling using this approach would make 3D modeling easy"*. An interesting application area focused on pedagogical tasks as such: *"...can be useful in applications for teaching, helping students visualize the objects and also to help physically challenged people to express their emotions and ideas"*.

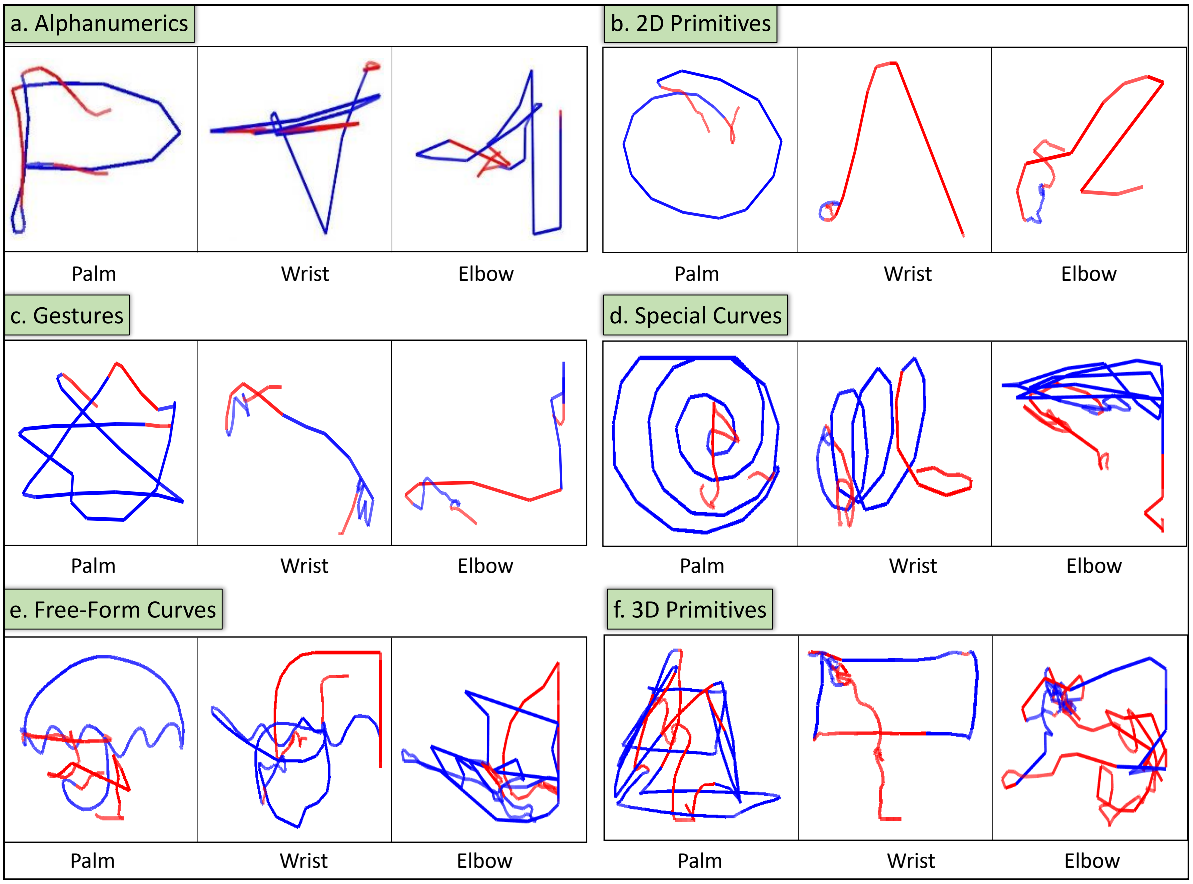


Figure 4.3: Variations in the mid-air spatial trajectories of the recorded palm, wrist, and elbow data for shapes drawn across the six categories.

4.2 Qualitative & Quantitative Analysis

Observations from the behavioral study indicate that for a given 3D hand trajectory, *stroke* and *hover* points exhibit interesting trends in speed profiles and average completion times. This section discusses about such trends observed in data collected across the 6 sessions using the custom hand-held device.

4.2.1 Visual Profiles

A quick visual analysis of the spatial profiles for the palm, wrist, and elbow trajectory indicate the following aspects. When drawing any given shape mid-air, the palm point travels maximum distance (Figure 4.4), while the elbow trajectory is observed to be almost stationary. It is observed that most participants used the elbow point as a pivot, and manipulated the wrist and palm to draw shapes. This observation is not very different from when a person writes on a piece of paper, or draws on a tablet. Spatial variations in the elbow trajectory are observed however, when 3D primitives are being drawn (Figure 4.3).

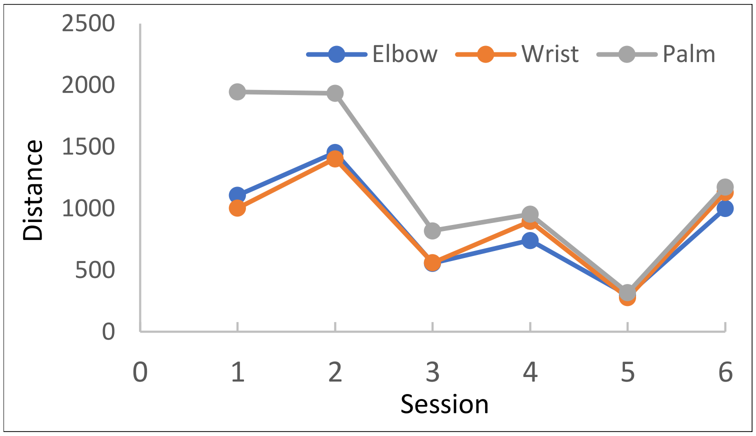


Figure 4.4: Average distance traversed by the palm, wrist, and elbow points across the six shape categories.

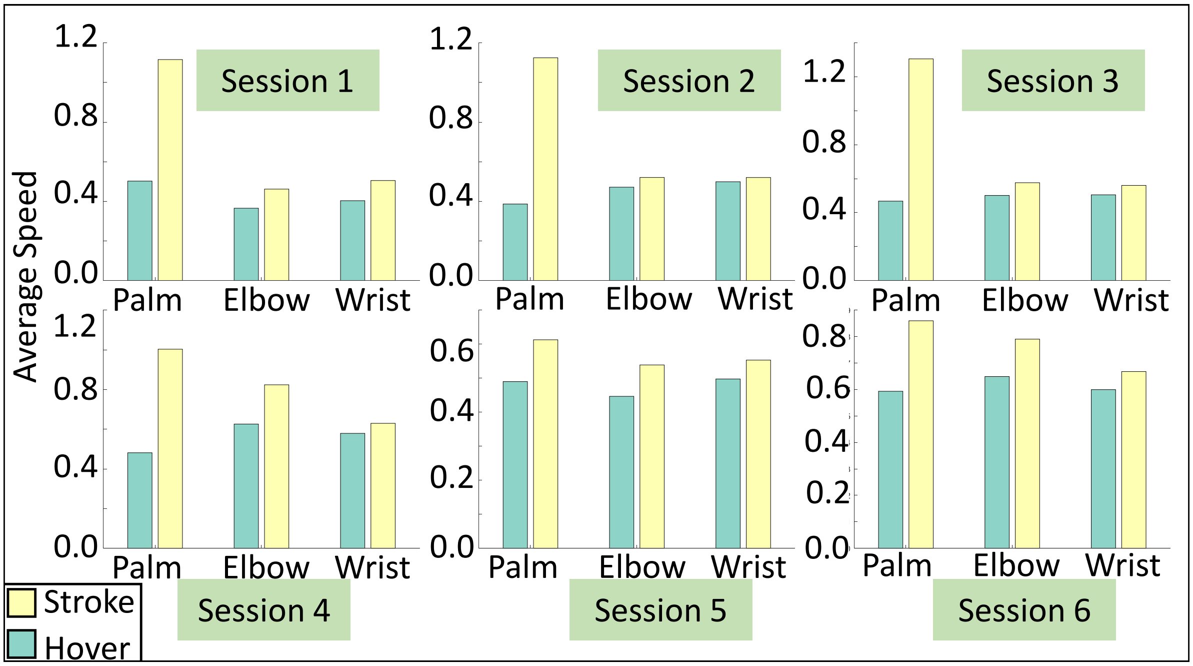


Figure 4.5: Average stroke-hover speeds of the palm-wrist-elbow trajectory for all drawn shapes across the six sessions.

4.2.2 Speed Profiles & Directionality

Overall *stroke-hover* speed profiles are different from as observed in the previous study. Here, on an average, strokes are traversed faster than hovers. Further, reduction in traversal speeds are observed along the transition points. Variations in speed profiles are observed across different shape categories (Figure 4.5). Alphanumeric characters, due to their familiarity, are traversed fastest (average speed $= 0.9 mm/s$), while free-form shapes are drawn the slowest (average speed $= 0.624$). Also, as observed in the behavioral study, *stroke* curves possess higher curvature (average curvature $= 0.096$) than *hover* curves, indicating that *hovers* are typically traversed in straight lines.

4.3 Classification Model Feature Design

4.3.1 Geometric Feature Based Models

Based on the mid-air drawing intent classification results discussed in the previous chapter, we first extract the following geometric properties from any given point on the recorded trajectories:

$$G_i = \begin{bmatrix} s_i & a_i & j_i & c_i & S_r & \omega_\alpha & \omega_\beta & \omega_\gamma \end{bmatrix} \tag{4.1}$$

where, s_i, a_i, j_i are the speed, acceleration, and jerk relative to the previous point; S_r is the ratio between speed of successive points; c_i is the local curvature; and ω_α, ω_β, and ω_γ represent the change in planarity of the recorded mid-air curve input.

These geometric features are used to construct the following models with the recorded data:

1. *One Point Geometric* G_1: Using the time-stamped coordinate of the palm trajectory and the geometric features above, we train a classifier using the following 8-dimensional geometric feature vector extracted from the recorded palm trajectory:

$$G_1 = \begin{bmatrix} s_{pi} & a_{pi} & j_{pi} & c_{pi} & S_{pr} & \omega_{p\alpha} & \omega_{p\beta} & \omega_{p\gamma} \end{bmatrix} \tag{4.2}$$

2. *Three Point Geometric* G_2 Along with considering the point-to-point geometric variations in the palm trajectory(G_p), we consider the wrist(G_w) and elbow (G_e) points too, and construct a 24-dimensional geometric feature vector as below:

$$G_2 = \begin{bmatrix} G_p & G_w & G_e \end{bmatrix} \tag{4.3}$$

4.3.2 Raw Data Based Local Differential Models

Geometric features described in the previous section are a derived representation of the nature of *stroke-hover* 3D data recorded. For any given shape drawn mid-air, the recorded palm-wrist-elbow data is a time stamped trajectory of sequential points in 3D. To extract the differentiating

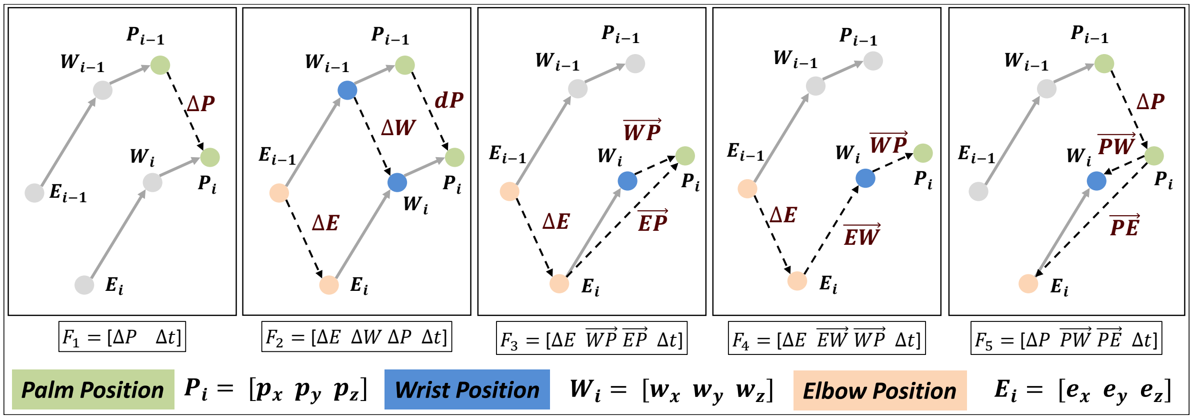

$F_1 = [\Delta P \;\; \Delta t]$
$F_2 = [\Delta E \;\; \Delta W \;\; \Delta P \;\; \Delta t]$
$F_3 = [\Delta E \;\; \overrightarrow{WP} \;\; \overrightarrow{EP} \;\; \Delta t]$
$F_4 = [\Delta E \;\; \overrightarrow{EW} \;\; \overrightarrow{WP} \;\; \Delta t]$
$F_5 = [\Delta P \;\; \overrightarrow{PW} \;\; \overrightarrow{PE} \;\; \Delta t]$
Palm Position $P_i = [p_x \; p_y \; p_z]$
Wrist Position $W_i = [w_x \; w_y \; w_z]$
Elbow Position $E_i = [e_x \; e_y \; e_z]$
P_{i-1}
W_{i-1}
E_{i-1}
P_i
W_i
E_i
ΔP
dP
ΔW
ΔE
$\overrightarrow{WP}$
$\overrightarrow{EP}$
$\overrightarrow{EW}$
$\overrightarrow{PW}$
$\overrightarrow{PE}$

Figure 4.6: One point and three point local differential representations for palm, wrist, and elbow data recorded.

inherent nature of the 3D data, we use the raw time-series coordinates to train classifier models. A typical sequential time series data may exhibit trends. To ensure these trends do not affect any computations, a common practice is to the difference the data at any instant. That is, the observation at time step t_i is represented as the difference with respect to observation at instance t_{i-1}. This removes the trend and the resultant difference series represents the changes in observations, in this case, changes in the 3D trajectories of the elbow, wrist, and palm when the user draws a shape mid-air.

For every given point at instance i,

$$P_i = \begin{bmatrix} P_{x,i} & P_{y,i} & P_{z,i} \end{bmatrix}, W_i = \begin{bmatrix} W_{x,i} & W_{y,i} & W_{z,i} \end{bmatrix}, E_i = \begin{bmatrix} E_{x,i} & E_{y,i} & E_{z,i} \end{bmatrix} \tag{4.4}$$

Five models based on different representations of the three trajectory points are constructed:

1. *One Point Representation (F_1)*: This is the simplest 4 dimensional vector (Figure 4.6) representation constructed using time stamped palm coordinates recorded for every user.

$$F_1 = \begin{bmatrix} \Delta P & \Delta t \end{bmatrix} \tag{4.5}$$

2. *Three Point Representations*: Using the 3-point tracking data, we embody the multi-joint

motion when drawing in mid-air through different representations of the bio-mechanical link system. While feature **F2** comprises of a simple difference between the palm, elbow, and wrist points between any two given instances, features **F3** and **F4** describe representations with elbow as the pivot for the bio-mechanical link. Feature **F5** considers a representation with the palm as the reference point for the consecutive wrist and elbow movements (Figure 4.6).

Feature F_2

$$F_2 = \begin{bmatrix} \Delta E & \Delta W & \Delta P & \Delta t \end{bmatrix} \tag{4.6}$$

Feature F_3

$$F_3 = \begin{bmatrix} \Delta E & \vec{WP} & \vec{EP} & \Delta t \end{bmatrix} \tag{4.7}$$

Feature F_4

$$F_4 = \begin{bmatrix} \Delta E & \vec{EW} & \vec{WP} & \Delta t \end{bmatrix} \tag{4.8}$$

Feature F_5

$$F_5 = \begin{bmatrix} \Delta P & \vec{PW} & \vec{PE} & \Delta t \end{bmatrix} \tag{4.9}$$

4.4 Training and Testing

A total of $165,000$ ($70,813$ *strokes*, $94,187$ *hovers*) time-stamped data points were recorded across the 6 shape categories from all participants. Experiments described in the previous chapter show random forests to be a comparatively better technique for estimating the *strokes*-*hovers* intent of recorded 3D data [65]. Of the total recorded data, $114,270$ points ($50,786$ *strokes*, $63,481$ *hovers*) are used for training the model, while the remaining $50,730$ points ($25,024$ *strokes*, $25,076$ *hovers*) are used to test and cross-validate (10-fold cross validation) the features. All models with feature representations **G1, G2** and **F1** to **F5** are trained using this data split. While training, the feature vectors were randomly sampled from the available pool to eliminate any kind of over-fitting of the trained model due to adjacent data points. Then, using the best feature representation, a hyper-parameter search for optimum random forest parameters is performed. All models are evaluated on the basis of their prediction accuracies, using the remaining 30% split of data. Prediction results for all 6 shape categories are finally discussed.

4.5 Results & Observations

In this section, the prediction accuracies for different models are discussed, and the best feature representation is identified. Initially, the random forest parameters are set to tree size $N_t = 40$, and default maximum splits. Next, a parametric optimization is performed to identify the best hyper-parameters for random forest with the chosen feature vector, and the best results for every shape category are discussed. Along with standard prediction accuracy (η), following additional metrics are used for comparing different feature models:

$$Precision = TP/(TP + FP), Recall = TP/(TP + FN)$$

$$TNR = TN/(TN + FP), TPR = TP/(TP + FN)$$

where, *TP*: True Positives; *TN*: True Negatives; *FP*: False Positives; *FN*: False Negatives; *TNR*: True Negative Rate; and *TPR*: True Positive Rate.

4.5.1 Feature Based Models

The classifiers trained using features G_1 and G_2 performed with an accuracy of $\eta = 72.33\%$ and $\eta = 73.1\%$ respectively. Testing accuracies from the palm-point feature G_1 align closely with the single point model discussed in the previous study, and exhibit a precision-recall rate of 0.659 and 0.616. The three point geometric feature G_3 predicts with a slightly better accuracy, however, both models exhibit a relatively high false negative rate.

4.5.2 Local Differential Models

4.5.2.1 One Point Model

With a 4-dimensional feature representation of the palm points(F1), the classifier results in a training accuracy of $\eta = 70.56\%$ and an average test accuracy of $\eta_{F1} = 76.07\%$. The classification results show a significant improvement in comparison with the real-time results obtained from the model trained using data collected on the haptic device. The results however, indicate a high degree of false negatives, with a precision-recall distribution of 0.733 and 0.692 respectively. This validates the need for high dimensional representation of the 3D data, which is discussed in the next section.

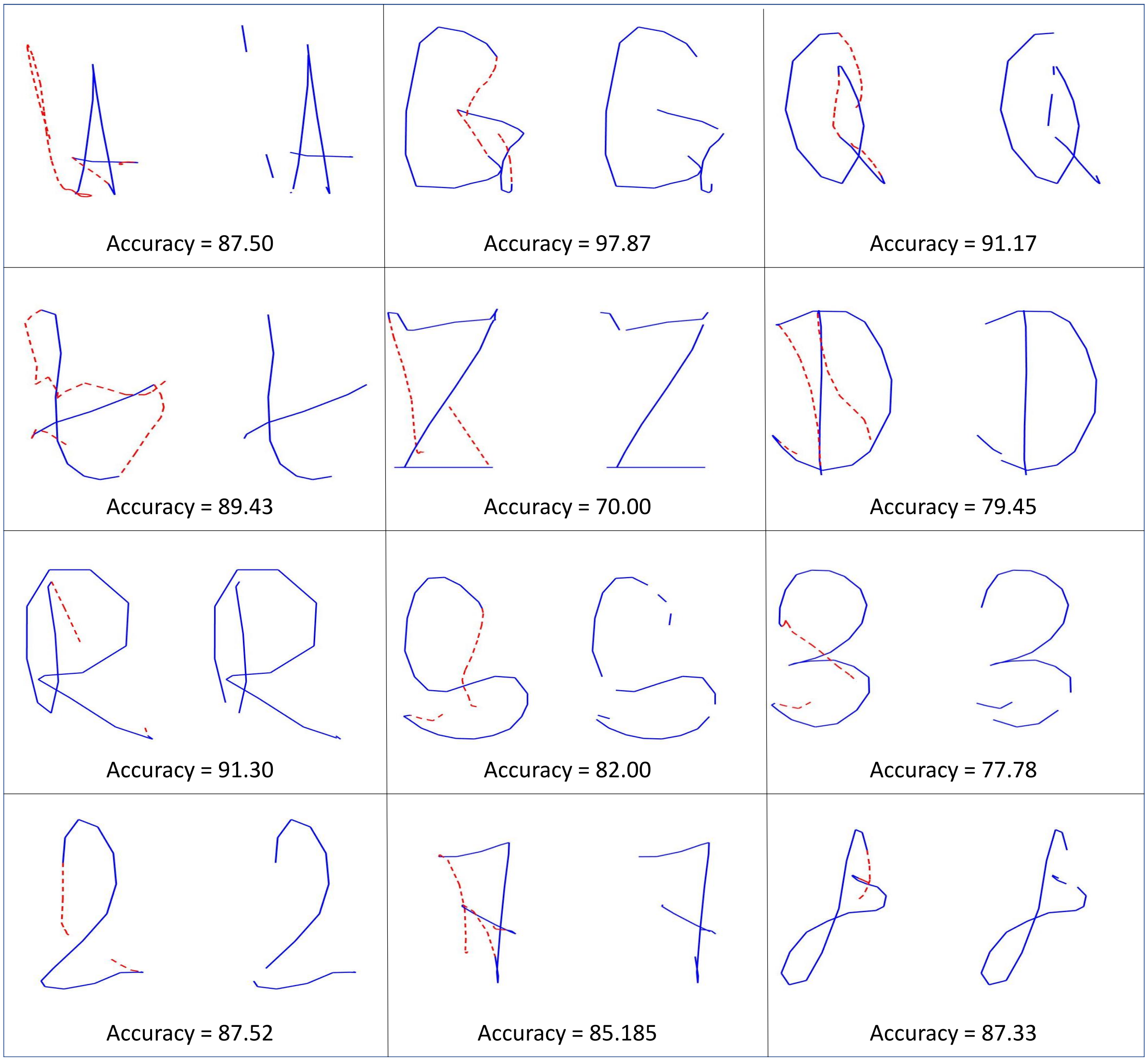


Figure 4.7: 3-point model Random Forest classifier prediction accuracies for alphanumeric sketch data recorded using the 3D Pen in Session 1.

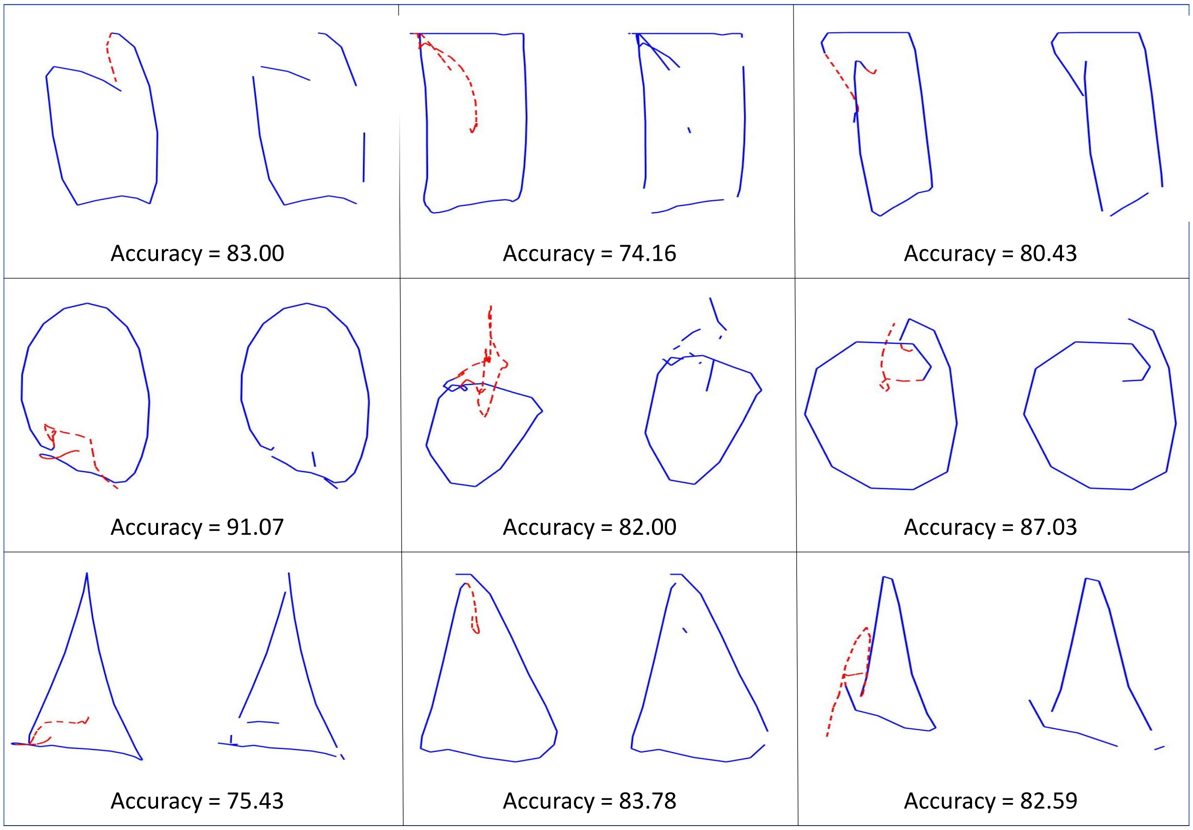


Figure 4.8: 3-point model Random Forest classifier prediction accuracies for 2D primitives recorded using the 3D Pen in Session 2.

4.5.2.2 Three Point Models

Remaining four data representations are trained and tested using a similar data split, as described before. Model 2 performs with a test accuracy $\eta_{F2} = 73.17\%$, whereas models 3 and 4 have average accuracies equal to $\eta_{F3} = 73.22\%$ and $\eta_{F4} = 73.26\%$ respectively. Model 5 on the other hand performs with the best accuracy of $\eta_{F5} = 79.17\%$. Further analysis of prediction accuracies across different sessions indicate that best results are obtained for shape categories 3 (motion gestures) and 4 (special curves), with $\eta = 80.33\%$ and $\eta = 81.36\%$ respectively. This feature representation is further used to identify best hyper-parameters of the classifier for optimum results.

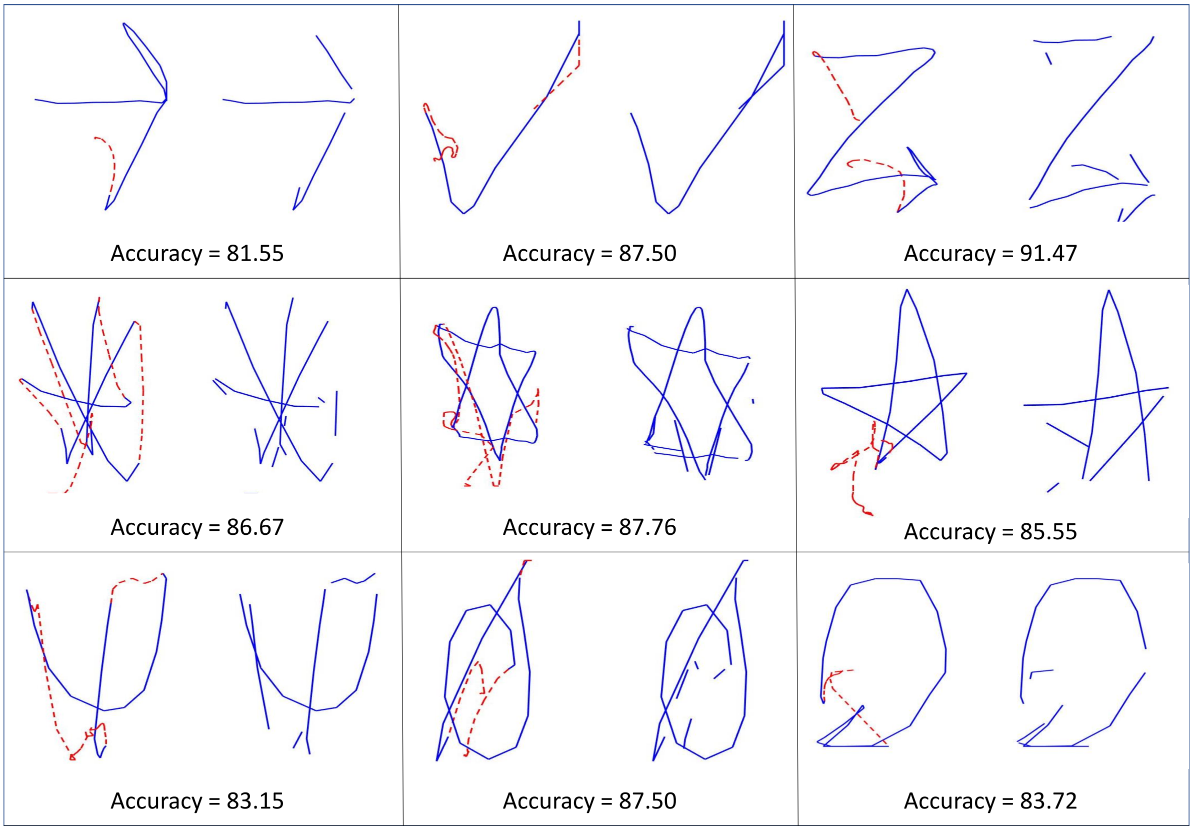


Figure 4.9: 3-point model Random Forest classifier prediction accuracies for gesture and motion curves recorded using the 3D Pen in Session 3.

4.5.3 Best Model & Hyper-parameter Exploration

Different models with varying number of trees (N_t) are trained using the best feature representation:

$$F_5 = \begin{bmatrix} \Delta P & \vec{PW} & \vec{PE} & \Delta t \end{bmatrix} \tag{4.10}$$

Starting with default values of $N_t = 10$, the trees are increased iteratively until $N_t = 150$. To counter the effects of *stroke-hover* imbalance in the data, a weighted cost matrix is used, with $c_{hover} = 1$ and $c_{stroke} = 1.15$. Based on this analysis, the model with $N_t = 60$ performs with an optimum overall accuracy of $\eta = 84.53\%$.

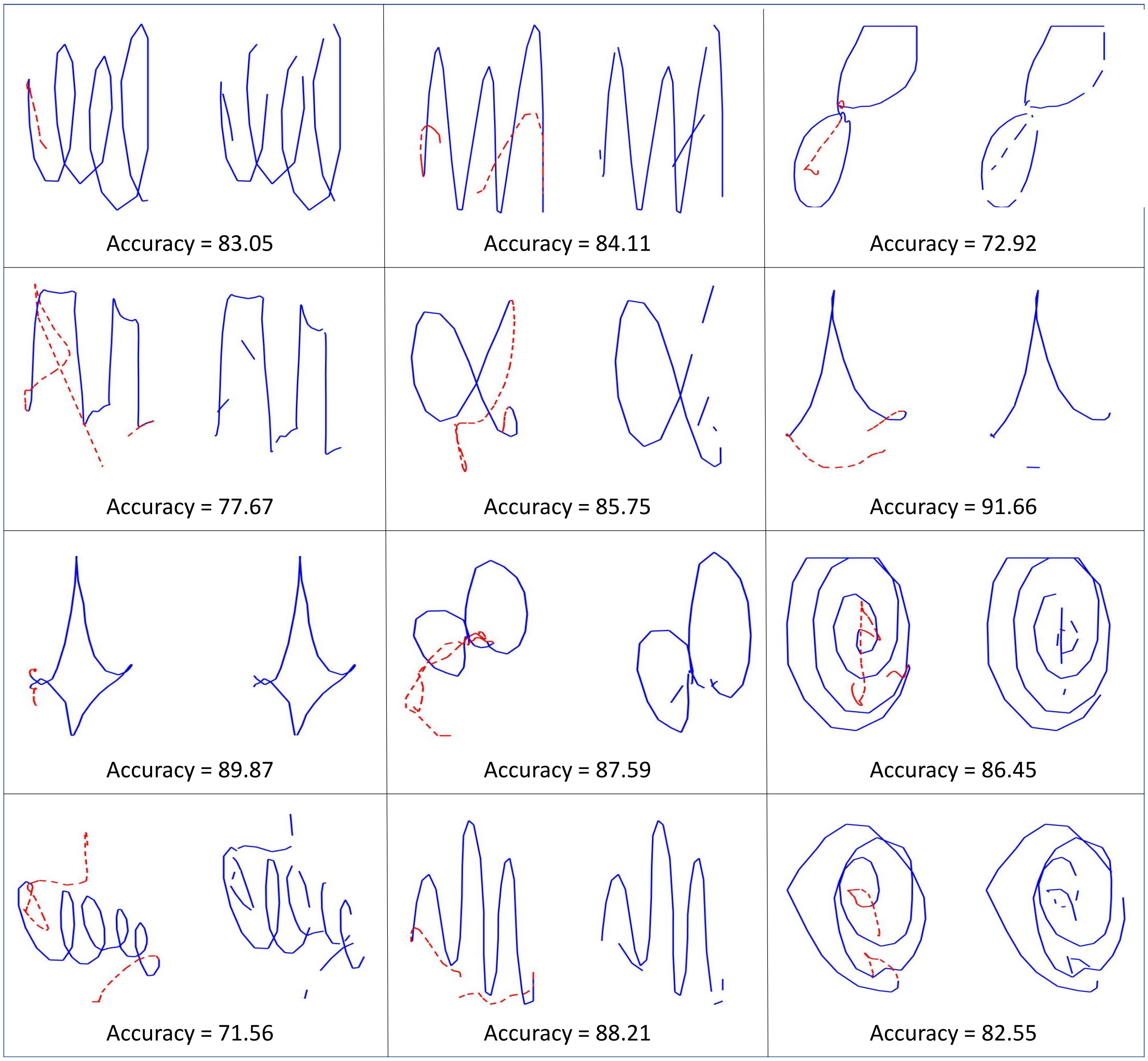


Figure 4.10: 3-point model Random Forest classifier prediction accuracies for special planar curves recorded using the 3D Pen in Session 4.

4.5.4 False Negative Trends

Across data from all 6 sessions, the predicted results are characterized by a true negative rate of $TNR = 0.8495$, while the true positive rate is observed as $TPR = 0.7655$. These metrics explain why the predicted *strokes* appear as fragments in certain predictions. Visual inspection of such false negatives point towards areas of high curvature, or high speed transitions. Incorporating an equal balance of such features in the *stroke-hover* dataset used for training can help improve these

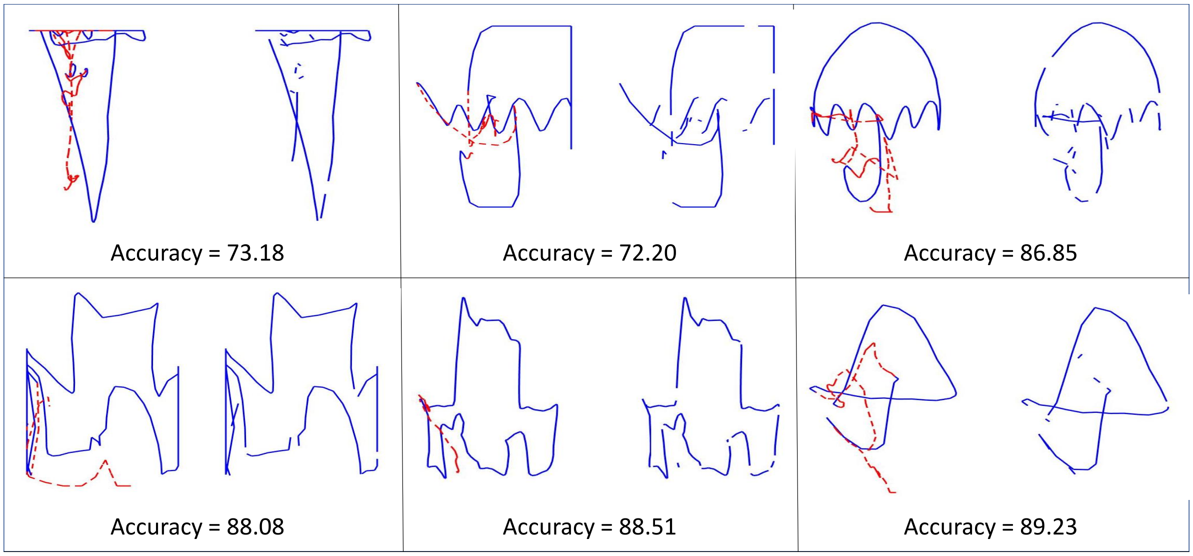


Figure 4.11: 3-point model Random Forest classifier prediction accuracies for free-form planar curves recorded using the 3D Pen in Session 5.

metrics, leading to better predictions.

4.5.5 Bare Hand Curve Data Predictions

To test how the best selected model performs with real-world data, a simple interface was developed on the OpenGL platform. The setup is maintained similar to the 3D pen data collection study described earlier, except that no hand-held Bluetooth connected device was used. The participants purely used their hand movements within Leap's interaction volume to draw shapes in mid-air. Data across the six categories was recorded. To observe how feature design affects the *stroke-hover* predictions, the recorded data was tested using all 4 models: geometric (G_1, G_2) and local differential (F_1, F_5). The 3-point differential model (F_5) clearly exhibits the best prediction results comparatively (Figure 4.13). While the intended negatives or *hovers* are appropriately identified, the predictions include false negatives too (Figure 4.14).

4.5.6 Live Prediction & Symbol Recognition Results

To test how the best selected model performs with real-world data, a simple interface was developed on the OpenGL platform. The setup was maintained similar to the Leap based data

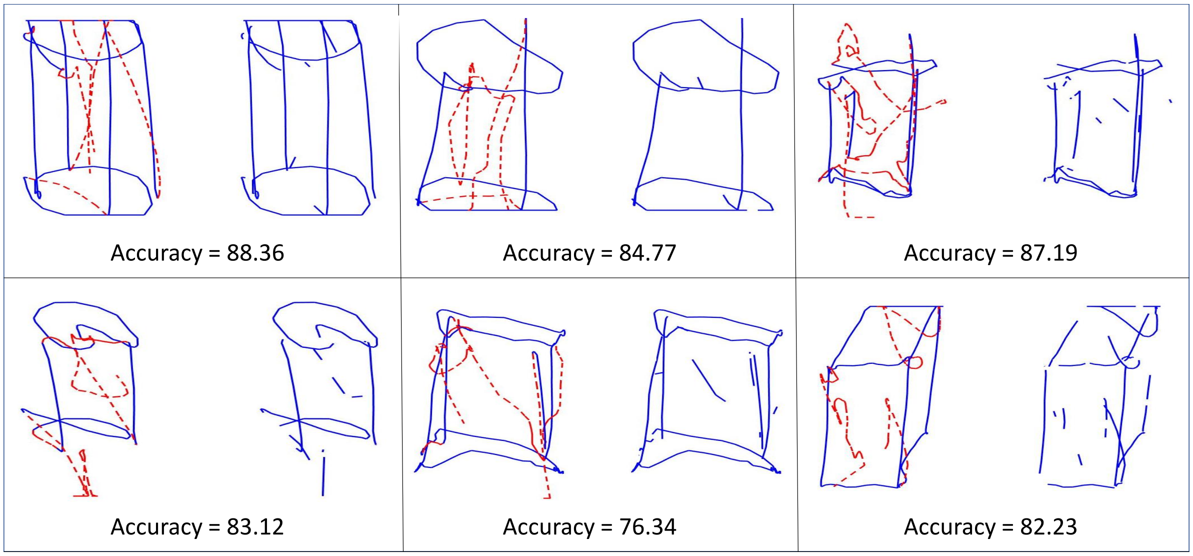


Figure 4.12: 3-point model Random Forest classifier prediction accuracies for 3D primitives recorded using the 3D Pen in Session 6.

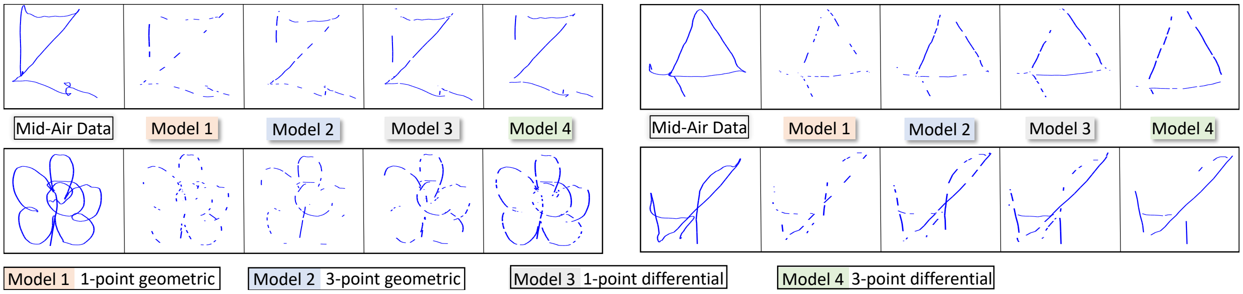
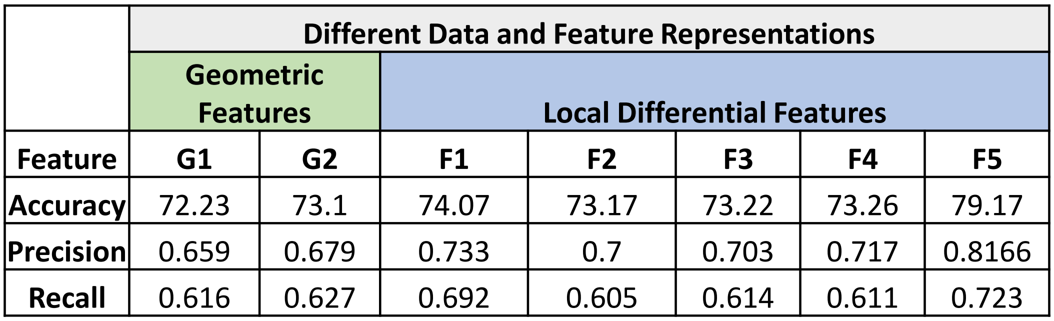

Mid-Air Data
Model 1
Model 2
Model 3
Model 4
Mid-Air Data
Model 1
Model 2
Model 3
Model 4
Model 1 1-point geometric
Model 2 3-point geometric
Model 3 1-point differential
Model 4 3-point differential

Figure 4.13: Comparison of stroke-hover prediction for bare-hand mid-air drawings by four different classifier models.

collection described earlier, except that no hand-held Bluetooth connected device was used. The participants purely used their hand movements within Leap's interaction volume to draw shapes in mid-air. The collected input was then passed onto an OpenCV implementation of Random Forests using optimum hyper parameters. The prediction results obtained are further processed using classic symbol recognition technique like the $\$P$ recognizer [21]. The recognizer is initially trained using templates sampled from the recorded data. Further exploration of the symbol recognizer pointed towards using the predicted shapes, instead of original user recorded shapes, for training the symbol recognizer.

The interface works with an overall accuracy of xx for yy number of total data points tested.



	Different Data and Feature Representations						
	Geometric Features		Local Differential Features				
Feature	G1	G2	F1	F2	F3	F4	F5
Accuracy	72.23	73.1	74.07	73.17	73.22	73.26	79.17
Precision	0.659	0.679	0.733	0.7	0.703	0.717	0.8166
Recall	0.616	0.627	0.692	0.605	0.614	0.611	0.723

Figure 4.14: Average accuracy, precision, and recall for test predictions using all combinations of geometric and differential feature vectors.

4.6 Conclusion

Despite eliminating spatial restrictions associated with the GeoMagic Touch device, the basic premise of the Leap controller data collection study involved using a hand-held device for recording the *stroke-hover* intent in mid-air. While the setup was designed to closely replicate how users typically draw in mid-air, the palm and wrist trajectories differ in the two scenarios, which is evident through the bare-hand Leap data predictions (Figure 4.13). In the future, it would be worthwhile to conduct a study with hand trackers and soft-buttons to make the effects of hand-held remotes less obvious during data collection. The biased false negatives predictions of the model dictates the need for identifying more critical geometric and temporal properties associated with *strokes*, leading to more accurate predictions. Due to the sensitivity of the predictive models towards the way data is represented, it is necessary to experiment with neural networks, that have the ability to learn the structure of data, without needing any specific features. Moreover, the data recorded in both studies involves asking the user to create *drawings* instead of *strokes*. A future study recording *strokes* only would help towards developing a better model for identifying the stroke-hover transition anomaly.

5. LATENT SPACE STROKE-HOVER CLASSIFICATION

Starting with insights from a behavioral study describing the way people draw in mid-air, we discussed two different types of models that help us with drawing intent classification. Results from the feature based model, and raw data representations indicate that the recorded mid-air curve consists of information that can be effectively used for training a binary classifier to identify the point-by-point *stroke-hover* intent. In this chapter, we explore the use of a special type of neural networks - autoencoders - to learn the *stroke-hover* characteristics from a latent embedding of the raw recorded mid-air curves. This chapter presents an introduction to autoencoders, and an overview of how autoencoders help capture important characteristics from the data through reconstruction. Further, we discuss a supervised learning methodology devised using autoencoders and random forests, and discuss some preliminary classification results obtained from this approach.

5.1 Autoencoders

An autoencoder is an artificial neural network that performs unsupervised learning on coded data [66]. Unlike algorithms discussed in the previous chapter, an autoencoder learns some form of encoded representation of the data in lower or higher dimensionality space. Autoencoders, like any other neural network do not need the input in an explicit "feature" space. They have been proven to learn patterns in raw image data sets effectively.

An autoencoder consists of two important units - the encoder and the decoder (Figure 5.1). At a minimum, a single autoencoder network consist of three layers: the input layer, the latent or hidden layer, and the output layer. The simplest autoencoder architecture is similar to a feed forward non-recurrent neural network, except that in an autoencoder, the input and output layers have the same dimensionality. When a certain input is passed to the first layer, the network's encoder encodes or compresses the data into a short code. The data in the encoded state is found in the hidden layer. Further, the autoencoder learns to decompress or decode this data back through the decoder to a representation as close as the original input data. This ultimately requires the autoencoder to

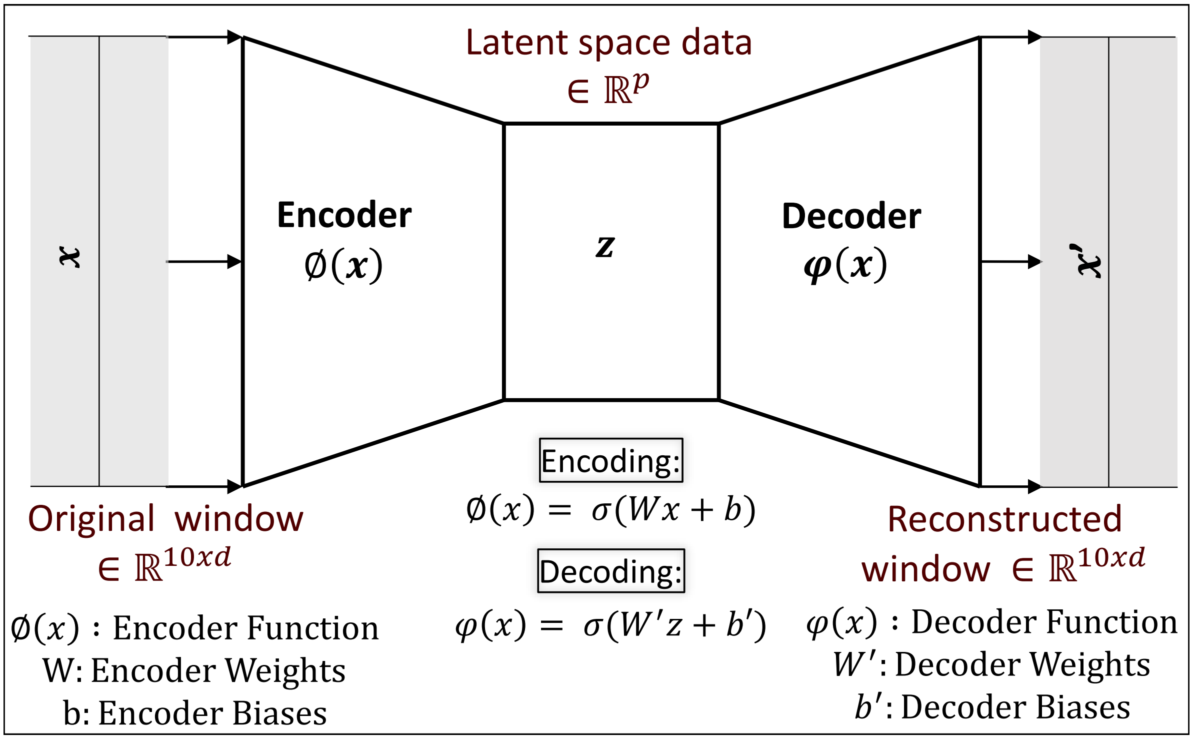


Figure 5.1: Architecture of a simple three layer autoencoder comprising of the encoder, decoder, and latent space.

perform dimensionality reduction on the data, and learn latent patterns in this lower dimensional embedding by reconstructing the data.

Mathematically, the autoencoder architecture with 3 layers can be explained as follows (Figure 5.1):

1. *Encoder*: The encoder or the first layer of the autoencoder takes in an input $x \in R^d = \chi$, and maps it using a transformation ϕ to $z \in R^p = \Gamma$ as $z = \sigma(Wx + b)$. The transformation z of the input data is referred to as the latent representation or latent variables. Here, σ is the element wise activation function, W is the weight matrix, and b is the vector or individual variable biases.

2. *Decoder*: This part then maps the latent variables (Γ) back to the input space (χ) through transformation ψ as $x' = \sigma'(W'z + b')$, where W' and b' are the weight and bias matrices for the decoder function ψ.

While reconstructing, autoencoders learn to minimize the reconstruction loss to ensure the x' matches as closely as possible with input x: $\mathcal{L}(x, x') = \|x - x'\|^2 = \|x - \sigma'(W'(\sigma(Wx + b)) + b')\|^2$

5.2 Rationale

Autoencoders are typically used for data dimension reduction by passing it through the encoder layer into the latent space. Several applications have proven the ability of autoencoders to identify clusters of data belonging to a given class, and use the information in the latent state to perform supervised and semi-supervised classifications. In context of *stroke-hover* binary classification problem, this research explores the utility of autoencoders combined with random forests to identify the drawing intent at a given point. This chapter explores the possibility of compressing the 3D drawing data to a latent space using autoencoders, and train a random forest classifier on this hidden space embedding of *strokes* and *hovers*.

5.3 Feature Space Formulation & Training

The construction and training of the autoencoder-random forest hybrid model is divided in two major tasks. The training data is distributed equally for these tasks:

1. *Autoencoder Training*: 3-point local differential representation of the training data (Features **F2** to **F5**) is used to train a simple three layer autoencoder. While learning how to reconstruct the data effectively, this autoencoder learns the latent space distribution of the data. We use window-based feature space representations of the training data, by concatenating *d* previous data points in the recorded time sequences. In other words, every feature vector for the autoencoder is a *10* $\times$ *d* dimensioned feature, with $d - 1$ feature points concatenated together. This *10* $\times$ *d* dimensioned feature is compressed to a hidden space of dimension *d* through the encoder function ϕ, which is typically the sigmoid function. The value of *d* is determined heuristically using hyper-parameter space exploration.

2. *Random Forest Training*: Once the autoencoder model is trained, the encoder part of the network is used to compress the second half of the training data into the latent space. This latent embedding of each data point is then used for a typical supervised learning classifi-

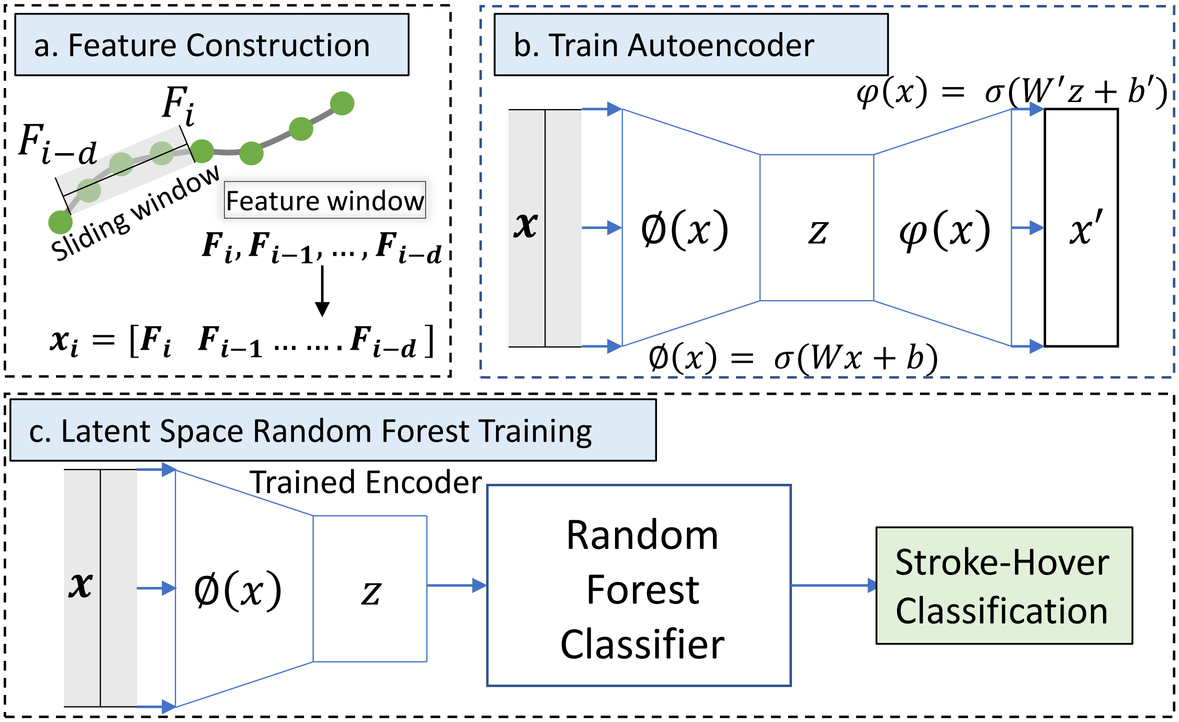

a. Feature Construction
F_i
F_{i-d}
Sliding window
Feature window
$F_i, F_{i-1}, \ldots, F_{i-d}$
$x_i = [F_i \;\; F_{i-1} \ldots\ldots F_{i-d}]$
b. Train Autoencoder
$\varphi(x) = \sigma(W'z + b')$
x
$\emptyset(x)$
z
$\varphi(x)$
x'
$\emptyset(x) = \sigma(Wx + b)$
c. Latent Space Random Forest Training
Trained Encoder
x
$\emptyset(x)$
z
Random Forest Classifier
Stroke-Hover Classification

Figure 5.2: Latent Space learning work-flow: a. Convert data to window based representation. b. Train the autoencoder network. c. Use the encoder of the trained network to transform recorded data to 3D space and train random forest model.

cation task to train the random forest model. The random forest parameters viz. number of trees (N_t) is initially equal to the best value obtained from the 3-point model trained earlier ($N_t = 60$).

Of the total $165,000$ data points, 70% were used for training, while the remaining 30% were used for testing. Half of the training data was used for training the autoencoder, and the remaining half for training the random forest.

For every new data point that is recorded mid-air, to identify the *stroke-hover* intent, following steps are followed:

1. *Latent Space Embedding*: For every feature of *10* $\times$ *d* dimension, the encoder part of the trained autoencoder compresses this feature to a latent space of dimension *d*.

2. *Random Forest Classification*: The encoded data from previous step is passed to the trained

random forest model, which classifies the feature as either a *stroke* or *hover*.

5.4 Preliminary Results

Features constructed from the remaining 30% of the dataset are used for testing the trained autoencoder-random forest model. While the results suggest that a latent space embedding of the 3D data helps with identifying the *stroke*-*hover* intent, the maximum test accuracy achieved is 72.09%, which is lower than the best 3-point model identified in the previous chapter. The average precision- recall for the entire test set is 0.6941 and 0.66819, whereas the true negative rate and true positive rates are calculated as $TNR = 0.803$ and $TPR = 0.606$ respectively. These metrics suggest that the hybrid model is further biased towards identifying *hovers* correctly, but is prone to mis-classify potential *strokes* as *hovers*.

5.5 Conclusion

This variation in the prediction accuracies for the latent space classification can be attributed to certain factors. Autoencoders, like any other neural network need a huge amount of data to learn the hidden patterns. With the total samples recorded from 25 participants, this data is insufficient to train an autoencoder, which can be seen from the high reconstruction error. This error is reflected in the form of a poor embedding of the latent space, which affects the way the random forest classifier is trained. Also, the original dimensionality of the data comprising of $P_{x,y,z}$, $W_{x,y,z}$, and $E_{x,y,z}$ is typically low for training an autoencoder - which have been found to perform exceptionally well while reconstructing images of 128×128 input dimension. Future studies that involve large amount of high dimensional data, like full upper body skeleton tracking, may help mitigate this problem.

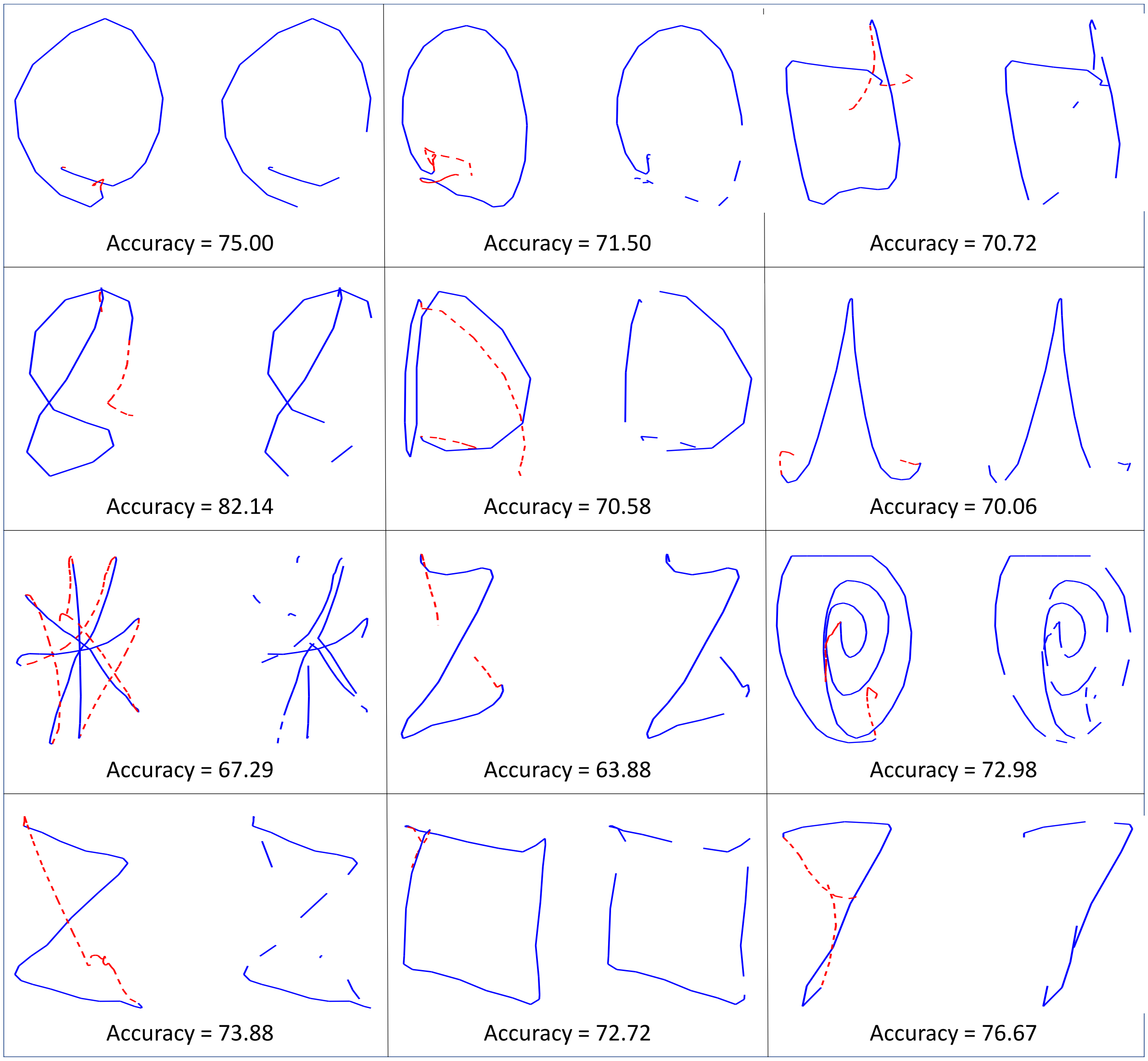
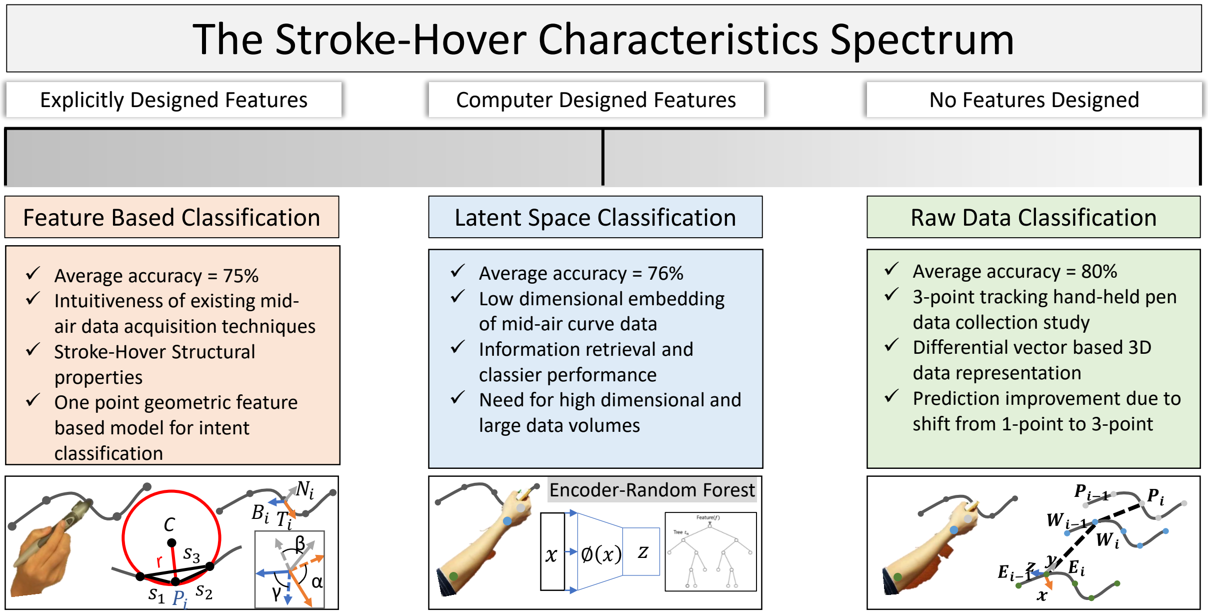


Figure 5.3: Stroke-Hover prediction results for different shape categories using autoencoder based latent space classifier.

6. CONCLUSION

This chapter concludes the thesis by first summarizing the technical and intellectual contributions, discussing implications of the proposed approach towards the design of spatial interactions, and finally discussing the potential areas for future work on mid-air intent recognition [1].



The Stroke-Hover Characteristics Spectrum

Explicitly Designed Features

Computer Designed Features

No Features Designed

Feature Based Classification

- ✓ Average accuracy = 75%
- ✓ Intuitiveness of existing mid-air data acquisition techniques
- ✓ Stroke-Hover Structural properties
- ✓ One point geometric feature based model for intent classification

Latent Space Classification

- ✓ Average accuracy = 76%
- ✓ Low dimensional embedding of mid-air curve data
- ✓ Information retrieval and classier performance
- ✓ Need for high dimensional and large data volumes

Raw Data Classification

- ✓ Average accuracy = 80%
- ✓ 3-point tracking hand-held pen data collection study
- ✓ Differential vector based 3D data representation
- ✓ Prediction improvement due to shift from 1-point to 3-point

Encoder-Random Forest

Figure 6.1: Summary of the key findings obtained from feature based, raw data based, and latent space based stroke-hover classification models.

6.1 Summary

This research presented a complementary approach for capturing user intent for creating strokes using spatial input. The idea was based on a simple premise that the user's intention to draw in mid-air can be implied through the motion and geometric characteristics of user-generated hand

[1] Part of the data reported in this chapter has been reprinted with permission from Association for Computing Machinery, Inc. and has been extracted from "To Draw or Not to Draw: Recognizing Stroke-Hover Intent in Non-instrumented Gesture-free Mid-Air Sketching" by Umema Bohari, Ting-Ju Chen, and Vinayak, In 23rd International Conference on Intelligent User Interfaces (IUI '18). ACM, New York, NY, USA, 177-188. DOI: https://doi.org/10.1145/3172944.3172985

trajectory. Feedback from the behavioral study provided key insights towards the need for developing such intent recognition methodology. We explored a spectrum of approaches through which *stroke-hover* characteristics can be represented (Figure 6.1). In the first feature based approach, we explicitly designed features to train the binary classifier. In the second raw data based approach, we explored the other end of the spectrum, where no explicit features were defined and we simply used raw data representations to classify the drawing intent. Finally, in the third latent space classification approach, we used autoencoders to help design the most appropriate features in the latent space, and used those features to identify the *stroke-hover* intent.

As the first step towards solving the *stroke-hover* intent recognition problem, we recorded 3D drawing data from different users through the GeoMagic Touch haptic device. Using the data recorded, we identified important geometric characteristics of *stroke-hover* points that can be used to train classification models. We explored three different binary classification algorithms, and identified random forests as the best model for this classification task. Finally, we discussed predictions from the optimum random forests models, and discussed some potential limitations with the approach.

To overcome the limitations with the feature based model, we conducted another study where we asked users to draw using a custom hand-held device, with their palm-wrist-elbow being tracked using the Leap controller. Instead of training the classifier on explicit features extracted from the recorded trajectories, we used a local differential representation of the raw data. By recording data encompassing multiple shape categories, the proposed 3-point model shows increase in accuracy in comparison to the features based model. In similar veins, a richer and more extensive dataset - primarily tracking the upper body skeleton - can be effectively used for training classifiers based on neural networks (especially recurrent neural networks that can capture the time-series nature of the underlying data).

Finally, in the third approach, using the 3-point model and autoencoders, we looked at identifying the user's drawing intentions from the latent space embedding of the processed data. Here, considering the mid-air trajectories as time sequences, we hypothesized that the drawing intent for a given point is dependent on the variation in the 3D position of its previous points. Using this assumption, we trained a hybrid autoencoder-random forests model to first compress each data vector to a lower dimensional space, and learn the latent *stroke-hover* characteristics of the data. While the average prediction accuracy for this hybrid model was lower as compared to the raw data based model, the results indicate that with a high dimensional data and large numbers of feature vectors for training, such an approach can give better insights for the *stroke-hover* classification problem.

6.2 Research Impact

At its core, the presented approach offers a different and complementary perspective for seamlessly processing mid-air curve inputs, that is, considering intentional strokes as *anomalies* within a continuous *hover* trajectory. The ability to detect this anomaly changes the way we design mid-air interfaces. In context of mid-air curve input, the primary idea behind the presented research is to minimize the use of predefined hand postures or instrumented controllers while drawing in mid-air. On completion, the implications of this work are manifold. First, by eliminating the use of gestures for intent identification, they can be used for other interface interaction tasks. Further, mid-air curves are the basis of a majority of applications dealing with 3D interactions. Generalization of this approach as an *intent* recognition task makes it applicable in applications ranging right from gesture tracking to activity recognition. Scaling this approach to multi-dimensional mid-air curve input comprising of hand tracking, body skeleton tracking, and gaze tracking trajectories could potentially improve the flexibility and ubiquity with which mid-air interactions are designed.

6.3 Discussion & Future Work

6.3.1 Mid-Air Data Dimensionality

Depending on the dimensionality of the data on which the classifier is trained, we observe variations in the prediction accuracy. The one point feature representation models in approach 1 and 2 (**F1**) trained using the time stamped stylus or palm coordinates predicted curves with an average accuracy much lower than the models trained using three point feature representations (**F2** to **F5**).

This shift can also be clearly seen in the reduction in false positives and false negatives, as one moves from one point to three point models - both, for feature based as well as raw geometric data representations (Figure 4.13). These variations suggest that apart from the palm trajectory, important *stroke-hover* information can be derived from the additional wrist and elbow joints. This observation is suggestive of the fact that though the overall movement of the wrist and elbow joints is smaller when compared to the palm, inclusion of higher dimensional data in the drawing intent classification task is an important aspect. In future studies, it would be worthwhile to understand how a person's movements of the upper body skeleton when drawing in mid-air affect the represented information on which the *stroke-hover* classification models are trained.

6.3.2 Data Representation for Stroke-Hover Intent Recognition

In this work, we experimented with two different ways in which information can be extracted from the recorded raw hand trajectories. First, in line with previous similar work [65], using both one point and three point representations, we extracted important geometric features from the raw data. While this feature based classifier was able to segregate the hovers, these models are characterized with a low precision-recall rate. On the other hand, the second data representation using simple local differences of the hand point trajectories exhibited high prediction accuracies as well as reasonable precision-recall. That is, modeling the variations in spatial coordinates of the time stamped hand-trajectories proved to be a good indicator of identifying a user's *stroke-hover* intent. Which brought us to this question: is there a way to extract *hidden characteristics* from *strokes* and *hovers*? Our third approach experimenting with autoencoders and the results so obtained suggest that in future, it would be worthwhile to experiment with unsupervised learning techniques specializing in time sequences, such as the recurrent neural networks.

6.3.3 Classifier Training Data Recording Modality

Training a classifier for drawing intent recognition task is a supervised learning problem – that is, it is necessary to record the mid-air drawing data with accurate *stroke-hover* demarcation. In the first approach, data was recorded using the movement of the 6-DOF stylus of the GeoMagic Touch haptic device for mid-air drawing tasks by several users. While the classifier trained using this data performed well for the haptic data predictions, it exhibited broken predictions and high false positives for bare hand data. In the second approach, we used the hand-held device with a Leap controller tracking the user's elbow-wrist-palm while they drew in mid-air. While this setup was designed to be as close as possible to the way people naturally would draw in mid-air, the inclusion of the hand-held device caused some variations in the wrist trajectories for both cases. The hand movements observed when using an instrumented controller, as compared to bare hand movements, are different. This explains the false positives and false negatives observed while predicting bare hand mid-air data. To ensure similarity between the trained classifier and its targeted application areas (say, mid-air drawing), it is necessary to use robust minimalistic hand posture based tracking for recording the raw data, or alternative modalities that would be less invasive to the user's hand movements in mid-air when drawing.

6.3.4 Robustness towards False Negatives

Bare hand leap data tested using all models proposed in this research exhibit a certain degree of false negatives. Typically localized in areas of high curvature, or high speed transitions, it would be interesting to record feature rich information for training the classifier. Bare hand mid-air drawing involves 6-DOF movements of the palm-wrist-elbow joint-link structure, ranging from simple translation, rotation about the joints, tilting, etc. Thus, along with recording the 3D coordinates of these joints, it is necessary to use accelerometers and gyroscopes to record these latter critical movements. A model trained on such data is expected to help improve the *stroke-hover* prediction metrics.

6.4 Applications

Some of the immediate implications of this work are discussed here.

6.4.1 Interaction Design

The validation of this model opens up new vistas in the domain of mid-air interactions in context of design ideation, and early stage concept design. With an on-the-fly implementation of the presented approach, designers will be able to naturally construct 3D concepts through consecutive curve descriptions without the need for specific gestures or instrumented controllers. The classified *stroke* data can then be recognized using standard sketch recognition techniques. Standard $N and $P recognition results discussed in the previous section indicate the utility of this approach in development of drawing applications based on 2D and 3D primitives, or any other templates that the recognizer is trained on. In collaborative design ideation setups, coupling this approach with semantic data may lead to the development of intelligent mixed-initiative interfaces.

Several works like Google's Quick, Draw! [64] and Sketch-RNN [67] discuss sketch recognition or sketch auto-completion applications using recurrent neural networks. The sketches or *strokes* are provided to the neural network through traditional tablet touch or mouse button-press based applications. Combination of the proposed intent recognition approach with such neural networks may lead to the development of suggestive mid-air interfaces, with the computer as a potential collaborator in cognitive tasks like design ideation and idea generation.

6.4.2 Extension Towards User Identification

The presented approach is simply based on the differential spatial data recorded from user's hand movements in mid air. Hayashi et al. [68] in their work *Wave2Me* discuss creation of authentic gestural and body length based signature IDs for multiple users on a shared network. An extension of the presented approach in similar directions may lead to development of a certain shape based ID for authentication purposes.

6.4.3 Development of Tangible Midair Feedback

A primary motivation behind this research has been minimizing the use of predefined postures or templates for intent recognition in mid-air drawing. Mid-air drawing in itself has seen tremendous developments over the last decade. With the advent of augmented and virtual reality, several devices simulating effective immersive environment have been developed. Applications like Google's Tilt Brush [28] and Gravity Sketch [69] allow experienced designers to create meaningful sketching/modeling artifacts in the 3D space. Along with the standard drawbacks associated with immersive environments, these applications lack the tangibility that is often necessary for drawing in mid-air. Combination of the proposed approach with a simple hand-held device capable of providing vibration or haptic feedback every time a stroke point is drawn, may lead to the development of interesting drawing applications with appropriate mid-air tactile feedback.

6.4.4 Applications to 3D Modeling

The presented approach can be considered complementary to existing segmentation and recognition applications. It is interesting to consider the implications of this model in context of 3-dimensional sketching applications like Tilt Brush [28] and Gravity Sketch [69], that use augmented reality and instrumented controllers for enabling mid-air sketching. Participant feedback from the behavioral study is indicative of the fact that sketching in mid-air without specifying explicit gestures, or using instrumented controllers, is both natural, as well as intuitive. The current work shows that both aspects are relevant and related. One of the major problems associated with using freehand 3D data in geometric modeling applications is the lack of controllability[36]. Scaling this approach and modeling geometric shape controllability using the *stroke-hover* analogy may lead to the development of mid-air curve input based 3D modeling applications.

Extending upon markerless hand tracking techniques used for object manipulation in 3D [43], this approach provides a starting point for context specific intent recognition, where the "intent" could be one of the many object manipulation tasks encountered in simpled CAD processes like assembly.

6.4.5 Scalability across different interaction volumes

In order for mid-air interactions to scale, we must have different levels of details for recognizing when a user wants to affect the state of an interactive system - specially in setups involving large displays. In this context, the presented approach offers a possibility of bypassing hand skeletal tracking by simply analyzing body-level activity (that can be robustly tracked by most available algorithms) to identify user intent for curve drawing [32]. This will further allow for implementation of novel and richer interfaces for large displays and interaction spaces enabling collaborative design experiences.

6.5 Conclusion

With the current accuracy of the proposed classification algorithms, it is possible to create interesting applications such as interactive art, especially for large displays with multiple users. Further, the validation of the approach in tandem with classic symbol recognition algorithms serves as a pointer to use this approach for developing intelligent mixed-initiative interfaces with the computer as a collaborator. In the broader domain of mid-air curve input, this work opens a new problem context that can potentially lead to novel approaches for enabling users to express visual ideas through spatial interactions.

6.6 Closing Statement

An important problem associated with bare hand mid-air interactions with the computer lies in extracting meaningful *intent* from the input curve, given the context of any specific application. To that extent, while several techniques have been developed that process the mid-air curves using external interrupts or segmentation based approaches, there is a necessity to explore more complementary approaches that treat the mid-air curve input in an as-natural-as-possible continuous point-by-point basis. With the *stroke-hover* classification approaches presented in this research serving as the starting point, there is a scope to develop scalable approaches applicable to general mid-air curve inputs for a variety of tasks. With the recent development of robust skeleton tracking cameras, developing high dimensional data models using entire upper body skeleton tracking can result in improvements in the performance of the context-specific intent prediction models.

REFERENCES

[1] J. Allen, C. I. Guinn, and E. Horvtz, “Mixed-initiative interaction,” *IEEE Intelligent Systems and their Applications*, vol. 14, no. 5, pp. 14–23, 1999.

[2] R. Arora, R. H. Kazi, F. Anderson, T. Grossman, K. Singh, and G. W. Fitzmaurice, “Experimental evaluation of sketching on surfaces in vr.,” in *CHI*, pp. 5643–5654, 2017.

[3] P. Taele, “Intelligent sketching interfaces for richer mid-air drawing interactions,” in *CHI’14 Extended Abstracts on Human Factors in Computing Systems*, (New York, NY, USA), pp. 339–342, ACM, 2014.

[4] Y. Chen, J. Liu, and X. Tang, “Sketching in the air: a vision-based system for 3d object design,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–6, IEEE, 2008.

[5] A. Schick, D. Morlock, C. Amma, T. Schultz, and R. Stiefelhagen, “Vision-based handwriting recognition for unrestricted text input in mid-air,” in *Proceedings of the 14th ACM international conference on Multimodal interaction*, pp. 217–220, ACM, 2012.

[6] S. Vikram, L. Li, and S. Russell, “Writing and sketching in the air, recognizing and controlling on the fly,” in *CHI’13 Extended Abstracts on Human Factors in Computing Systems*, pp. 1179–1184, ACM, 2013.

[7] C. Agarwal, D. P. Dogra, R. Saini, and P. P. Roy, “Segmentation and recognition of text written in 3d using leap motion interface,” in *Pattern Recognition (ACPR), 2015 3rd IAPR Asian Conference on*, pp. 539–543, IEEE, 2015.

[8] I. Aslan, A. Uhl, A. Meschtscherjakov, and M. Tscheligi, “Mid-air authentication gestures: an exploration of authentication based on palm and finger motions,” in *Proceedings of the 16th International Conference on Multimodal Interaction*, (New York, NY, USA), pp. 311–318, ACM, 2014.

[9] E. M. Taranta II, A. Samiei, M. Maghoumi, P. Khaloo, C. R. Pittman, and J. J. LaViola Jr., “Jackknife: A reliable recognizer with few samples and many modalities,” in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI ’17, (New York, NY, USA), pp. 5850–5861, ACM, 2017.

[10] P. Suryanarayan, A. Subramanian, and D. Mandalapu, “Dynamic hand pose recognition using depth data,” in *Proceedings of the 2010 20th International Conference on Pattern Recognition*, ICPR ’10, (Washington, DC, USA), pp. 3105–3108, IEEE Computer Society, 2010.

[11] R. ArandjeloviÄĞ and T. M. Sezgin, “Sketch recognition by fusion of temporal and image-based features,” *Pattern Recognition*, vol. 44, no. 6, pp. 1225 – 1234, 2011.

[12] D. Willems, R. Niels, M. van Gerven, and L. Vuurpijl, “Iconic and multi-stroke gesture recognition,” *Pattern Recognition*, vol. 42, no. 12, pp. 3303 – 3312, 2009. New Frontiers in Handwriting Recognition.

[13] Z. Ren, J. Meng, and J. Yuan, “Depth camera based hand gesture recognition and its applications in human-computer-interaction,” in *2011 8th International Conference on Information, Communications Signal Processing*, pp. 1–5, IEEE, Dec. 2011.

[14] F. Dominio, M. Donadeo, and P. Zanuttigh, “Combining multiple depth-based descriptors for hand gesture recognition,” *Pattern Recogn. Lett.*, vol. 50, pp. 101–111, Dec. 2014.

[15] N. C. Krishnan and D. J. Cook, “Activity recognition on streaming sensor data,” *Pervasive Mob. Comput.*, vol. 10, pp. 138–154, Feb. 2014.

[16] C. Holz and A. Wilson, “Data miming: Inferring spatial object descriptions from human gesture,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’11, (New York, NY, USA), pp. 811–820, ACM, 2011.

[17] Vinayak and K. Ramani, “A gesture-free geometric approach for mid-air expression of design intent in 3d virtual pottery,” *Computer-Aided Design*, vol. 69, pp. 11 – 24, 2015.

[18] S.-H. Bae, R. Balakrishnan, and K. Singh, “Ilovesketch: As-natural-as-possible sketching system for creating 3d curve models,” in *Proceedings of the 21st Annual ACM Symposium*

on User Interface Software and Technology, UIST '08, (New York, NY, USA), pp. 151–160, ACM, 2008.

[19] J. Dorsey, S. Xu, G. Smedresman, H. Rushmeier, and L. McMillan, "The mental canvas: A tool for conceptual architectural design and analysis," in *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, PG '07, (Washington, DC, USA), pp. 201–210, IEEE Computer Society, 2007.

[20] L. Anthony and J. O. Wobbrock, "A lightweight multistroke recognizer for user interface prototypes," in *Proceedings of Graphics Interface 2010*, GI '10, (Toronto, Ont., Canada, Canada), pp. 245–252, Canadian Information Processing Society, 2010.

[21] L. Anthony and J. O. Wobbrock, "$n-protractor: A fast and accurate multistroke recognizer," in *Proceedings of Graphics Interface 2012*, GI '12, (Toronto, Ont., Canada, Canada), pp. 117–120, Canadian Information Processing Society, 2012.

[22] J. O. Wobbrock, A. D. Wilson, and Y. Li, "Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes," in *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, (New York, NY, USA), pp. 159–168, ACM, 2007.

[23] R.-D. Vatavu, L. Anthony, and J. O. Wobbrock, "Gestures as point clouds: A $p recognizer for user interface prototypes," in *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, ICMI '12, (New York, NY, USA), pp. 273–280, ACM, 2012.

[24] P. Lapides, E. Sharlin, M. C. Sousa, and L. Streit, "The 3d tractus: A three-dimensional drawing board," in *Horizontal Interactive Human-Computer Systems, 2006. TableTop 2006. First IEEE International Workshop on*, pp. 8–pp, IEEE, 2006.

[25] M. F. Deering, "Holosketch: a virtual reality sketching/animation tool," *ACM Transactions on Computer-Human Interactions*, vol. 2, no. 3, pp. 220–238, 1995.

[26] D. F. Keefe, D. A. Feliz, T. Moscovich, D. H. Laidlaw, and J. J. LaViola, Jr., "Cavepainting: a fully immersive 3d artistic medium and interactive experience," in *Proceedings of the ACM symposium on Interactive 3D graphics*, pp. 85–93, 2001.

[27] M. Xin, E. Sharlin, and M. C. Sousa, "Napkin sketch: Handheld mixed reality 3d sketching," in *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology*, VRST '08, (New York, NY, USA), pp. 223–226, ACM, 2008.

[28] Google, "Tilt Brush." https://www.tiltbrush.com/, 2017. [Online; accessed 01-October-2017].

[29] E. Sachs, A. Roberts, and D. Stoops, "3draww: A tool for designing 3d shapes," *IEEE Comput. Graph. Appl.*, vol. 11, pp. 18–26, Nov. 1991.

[30] "Manipulation aid for two-handed 3-d designing within a shared virtual environment,"

[31] T. Grossman, R. Balakrishnan, G. Kurtenbach, G. Fitzmaurice, A. Khan, and B. Buxton, "Creating principal 3d curves with digital tape drawing," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, (New York, NY, USA), pp. 121–128, ACM, 2002.

[32] T. Grossman, R. Balakrishnan, G. Kurtenbach, G. Fitzmaurice, A. Khan, and B. Buxton, "Interaction techniques for 3d modeling on large displays," in *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, I3D '01, (New York, NY, USA), pp. 17–23, ACM, 2001.

[33] R. Balakrishnan, G. Fitzmaurice, G. Kurtenbach, and W. Buxton, "Digital tape drawing," in *Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology*, UIST '99, (New York, NY, USA), pp. 161–169, ACM, 1999.

[34] B. Laundry, M. Masoodian, and B. Rogers, "Interaction with 3d models on large displays using 3d input techniques," in *Proceedings of the 11th International Conference of the NZ Chapter of the ACM Special Interest Group on Human-Computer Interaction*, CHINZ '10, (New York, NY, USA), pp. 49–56, ACM, 2010.

[35] S. Schkolne, M. Pruett, and P. Schröder, "Surface drawing: creating organic 3d shapes with the hand and tangible tools," in *Proceedings of the ACM conference on Human factors in computing systems*, pp. 261–268, 2001.

[36] D. Keefe, R. Zeleznik, and D. Laidlaw, "Drawing on air: Input techniques for controlled 3d line illustration," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, pp. 1067–1081, Sept 2007.

[37] T. Ni, G. S. Schmidt, O. G. Staadt, M. A. Livingston, R. Ball, and R. May, "A survey of large high-resolution display technologies, techniques, and applications," in *Proceedings of the IEEE Conference on Virtual Reality*, VR '06, (Washington, DC, USA), pp. 223–236, IEEE Computer Society, 2006.

[38] M. Czerwinski, G. Robertson, B. Meyers, G. Smith, D. Robbins, and D. Tan, "Large display research overview," in *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, (New York, NY, USA), pp. 69–74, ACM, 2006.

[39] L. Lischke, J. Grüninger, K. Klouche, A. Schmidt, P. Slusallek, and G. Jacucci, "Interaction techniques for wall-sized screens," in *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*, ITS '15, (New York, NY, USA), pp. 501–504, ACM, 2015.

[40] S. K. Behera, P. Kumar, D. P. Dogra, and P. P. Roy, "Fast signature spotting in continuous air writing," in *Machine Vision Applications (MVA), 2017 Fifteenth IAPR International Conference on*, pp. 314–317, IEEE, 2017.

[41] M. G. Jacob and J. P. Wachs, "Context-based hand gesture recognition for the operating room," *Pattern Recognition Letters*, vol. 36, pp. 196 – 203, 2014.

[42] J. Segen and S. Kumar, "Gesture vr: vision-based 3d hand interace for spatial interaction," in *Proceedings of the sixth ACM international conference on Multimedia*, pp. 455–464, ACM, 1998.

[43] R. Wang, S. Paris, and J. Popović, “6d hands: markerless hand-tracking for computer aided design,” in *Proceedings of the ACM symposium on User interface software and technology*, pp. 549–558, 2011.

[44] Vinayak, S. Murugappan, H. Liu, and K. Ramani, “Shape-it-up: Hand gesture based creative expression of 3d shapes using intelligent generalized cylinders,” *Computer-Aided Design*, vol. 45, no. 2, pp. 277–287, 2013.

[45] L. W. Campbell and A. F. Bobick, “Recognition of human body motion using phase space constraints,” in *Computer vision, 1995. proceedings., fifth international conference on*, pp. 624–630, IEEE, 1995.

[46] J. Yamato, J. Ohya, and K. Ishii, “Recognizing human action in time-sequential images using hidden markov model,” in *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR’92., 1992 IEEE Computer Society Conference on*, pp. 379–385, IEEE, 1992.

[47] M. Brand, N. Oliver, and A. Pentland, “Coupled hidden markov models for complex action recognition,” in *Computer vision and pattern recognition, 1997. proceedings., 1997 ieee computer society conference on*, pp. 994–999, IEEE, 1997.

[48] H.-K. Lee and J.-H. Kim, “An hmm-based threshold model approach for gesture recognition,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 10, pp. 961–973, 1999.

[49] C. Rao and M. Shah, “View-invariance in action recognition,” in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 2, pp. II–II, IEEE, 2001.

[50] J. Aggarwal and L. Xia, “Human activity recognition from 3d data: A review,” *Pattern Recognition Letters*, vol. 48, pp. 70 – 80, 2014. Celebrating the life and work of Maria Petrou.

[51] H. Junker, O. Amft, P. Lukowicz, and G. TrÃűster, “Gesture spotting with body-worn inertial sensors to detect user activities,” *Pattern Recognition*, vol. 41, no. 6, pp. 2010 – 2024, 2008.

[52] O. Cakmakci, J. Coutaz, K. Van Laerhoven, and H.-W. Gellersen, “Context awareness in systems with limited resources,” *context*, vol. 2, no. 1, p. 2ln, 2002.

[53] P. Lukowicz, J. A. Ward, H. Junker, M. Stäger, G. Tröster, A. Atrash, and T. Starner, “Recognizing workshop activity using body worn microphones and accelerometers,” in *International conference on pervasive computing*, pp. 18–32, Springer, 2004.

[54] O. D. Lara and M. A. Labrador, “A survey on human activity recognition using wearable sensors.,” *IEEE Communications Surveys and Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013.

[55] G. Johnson, M. D. Gross, J. Hong, and E. Yi-Luen Do, “Computational support for sketching in design: A review,” *Found. Trends Hum.-Comput. Interact.*, vol. 2, pp. 1–93, Jan. 2009.

[56] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine,” in *International workshop on ambient assisted living*, pp. 216–223, Springer, 2012.

[57] MathWorks, “Statistics and Machine Learning Toolbox.” https://www.mathworks.com/products/statistics.html. [Online; accessed 01-October-2017].

[58] N. S. Altman, “An introduction to kernel and nearest-neighbor nonparametric regression,” *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[59] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, pp. 273–297, Sep 1995.

[60] Wikipedia, “k-nearest neighbor algorithm.” https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm. [Online; accessed 01-October-2017].

[61] T. Hastie, R. Tibshirani, and J. Friedman, “The elements of statistical learning. 2001,” 2001.

[62] Wikipedia, “Support Vector Machine.” https://en.wikipedia.org/wiki/Support_vector_machine. [Online; accessed 01-October-2017].

[63] Wikipedia, “Random Forest.” https://en.wikipedia.org/wiki/Random_forest. [Online; accessed 01-October-2017].

[64] Google, "Quick, Draw!." https://quickdraw.withgoogle.com/. [Online; accessed 01-October-2017].

[65] U. Bohari, T.-J. Chen, *et al.*, "To draw or not to draw: Recognizing stroke-hover intent in non-instrumented gesture-free mid-air sketching," in *23rd International Conference on Intelligent User Interfaces*, pp. 177–188, ACM, 2018.

[66] Wikipedia, "Autoencoder." https://en.wikipedia.org/wiki/Autoencoder. [Online; accessed 01-October-2017].

[67] I. J. David Ha, Jonas Jongejan, "sketch-rnn." https://experiments.withgoogle.com/sketch-rnn-demo. [Online; accessed 01-October-2017].

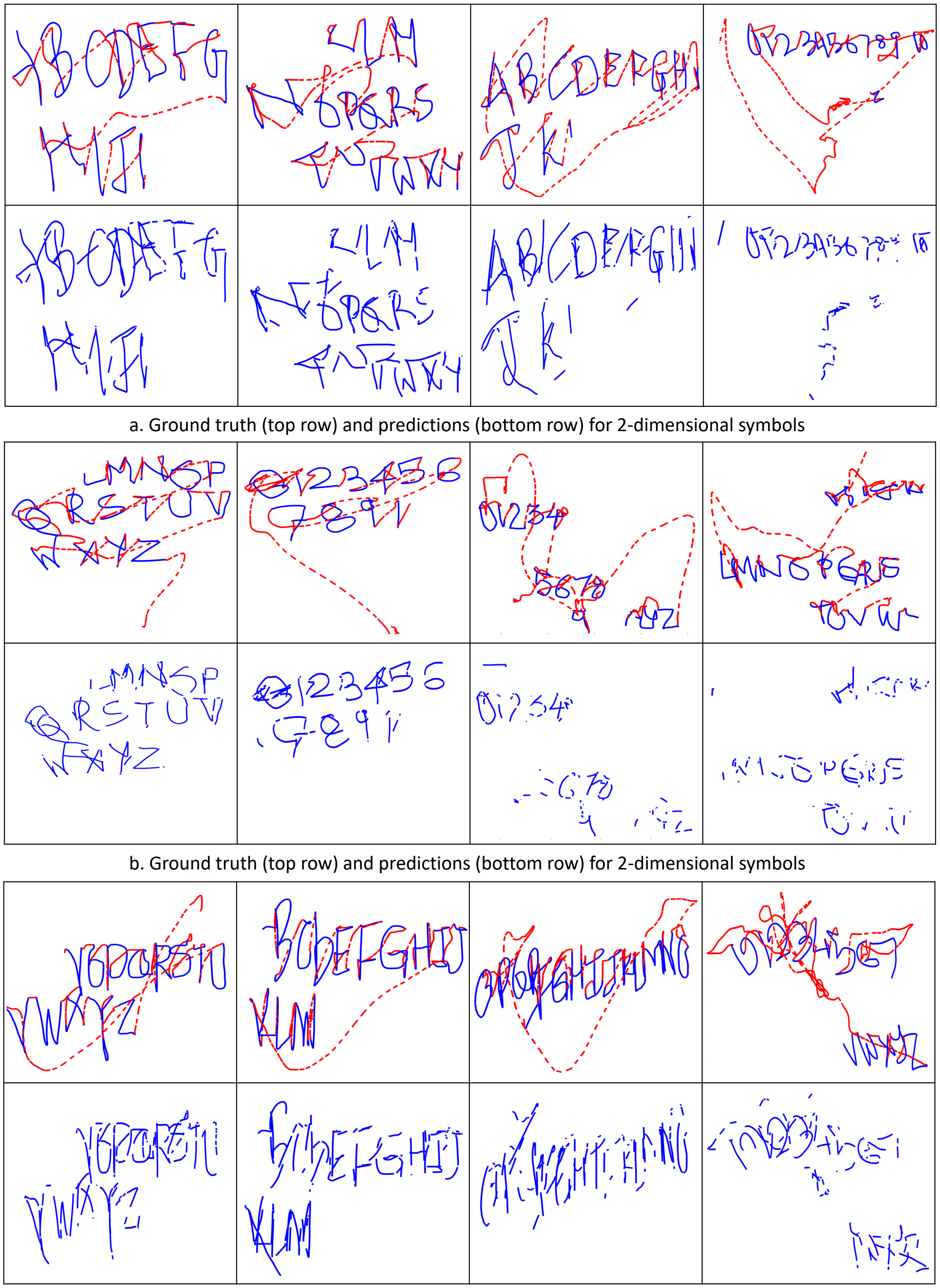
[68] E. Hayashi, M. Maas, and J. I. Hong, "Wave to me: user identification using body lengths and natural gestures," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 3453–3462, ACM, 2014.

[69] G. Sketch, "Gravity Sketch." https://www.gravitysketch.com/, 2017. [Online; accessed 01-October-2017].

APPENDIX A

FEATURE BASED CLASSIFICATION RANDOM FOREST PREDICTIONS

In this section, we show all random forest model prediction results for symbols, 2D and 3D primitives, and free-form sketches sampled from the 50000 testing data points recorded across the 19 participants.

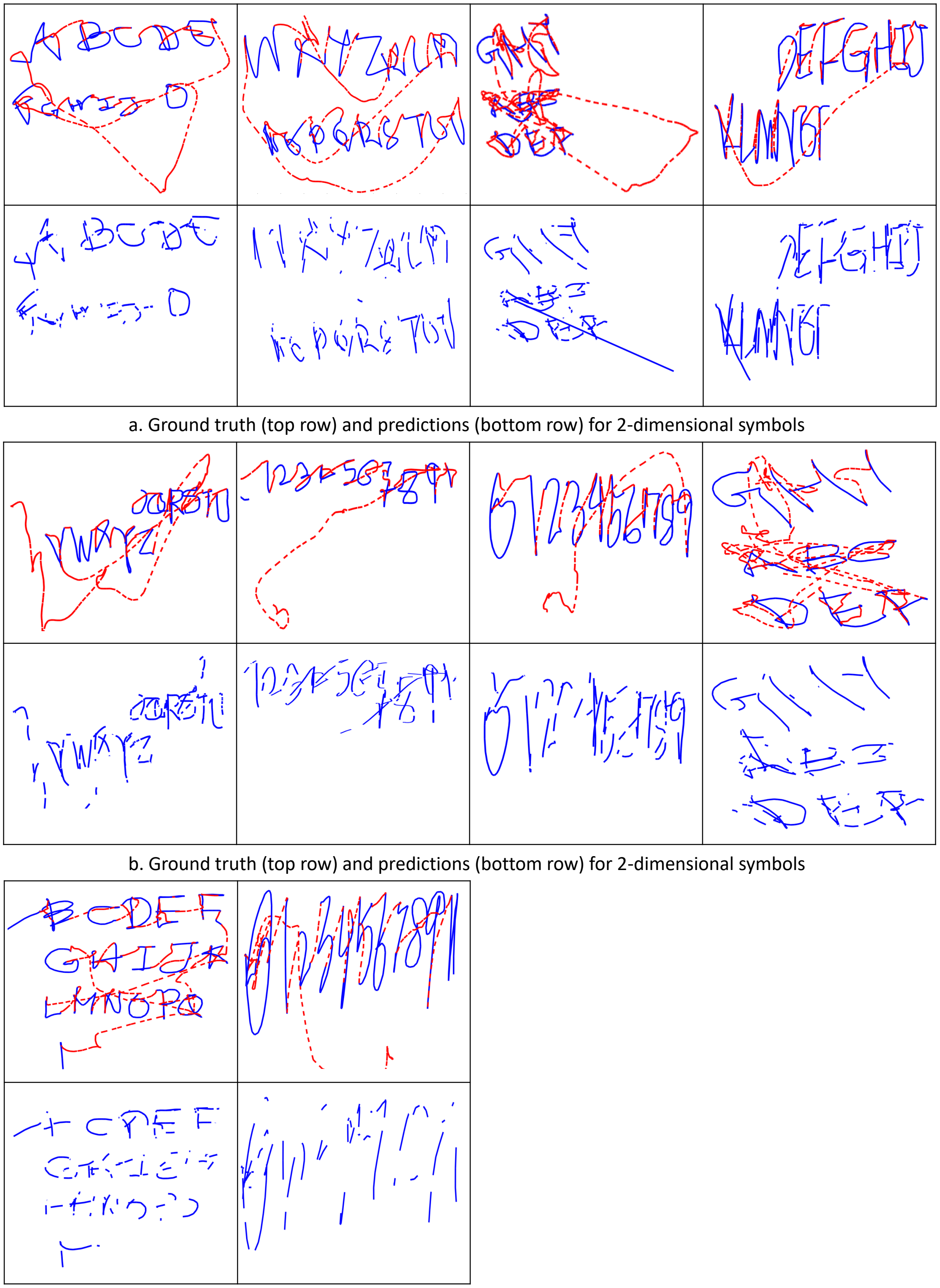


a. Ground truth (top row) and predictions (bottom row) for 2-dimensional symbols

b. Ground truth (top row) and predictions (bottom row) for 2-dimensional symbols

c. Ground truth (top row) and predictions (bottom row) for 2-dimensional symbols

Figure A.1: Random Forest model predictions for symbols data recorded using GeoMagic Touch.

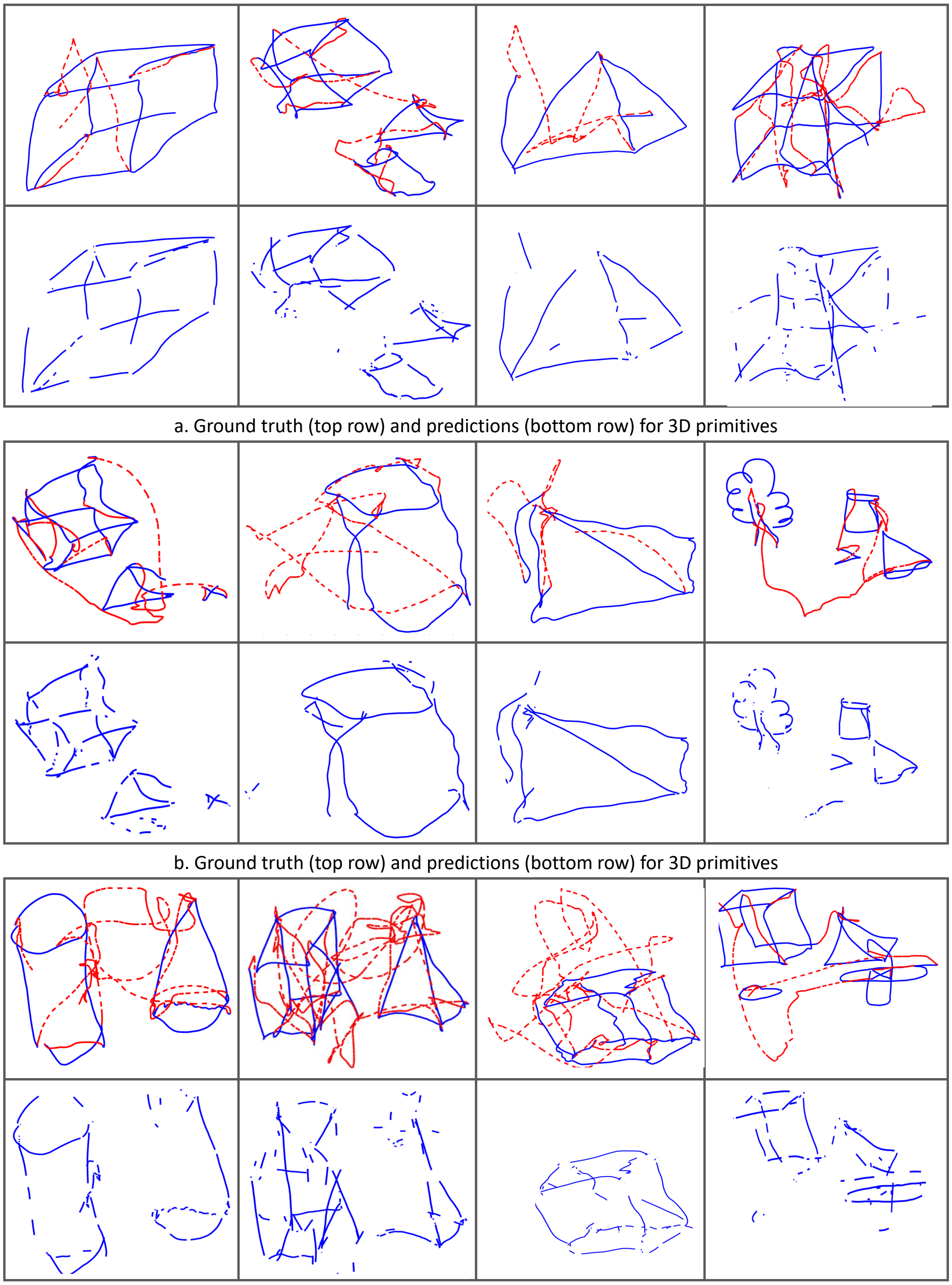


a. Ground truth (top row) and predictions (bottom row) for 2-dimensional symbols

b. Ground truth (top row) and predictions (bottom row) for 2-dimensional symbols

c. Ground truth (top row) and predictions (bottom row) for 2-dimensional symbols

Figure A.2: Random Forest model predictions for alphanumerics data recorded using GeoMagic Touch.

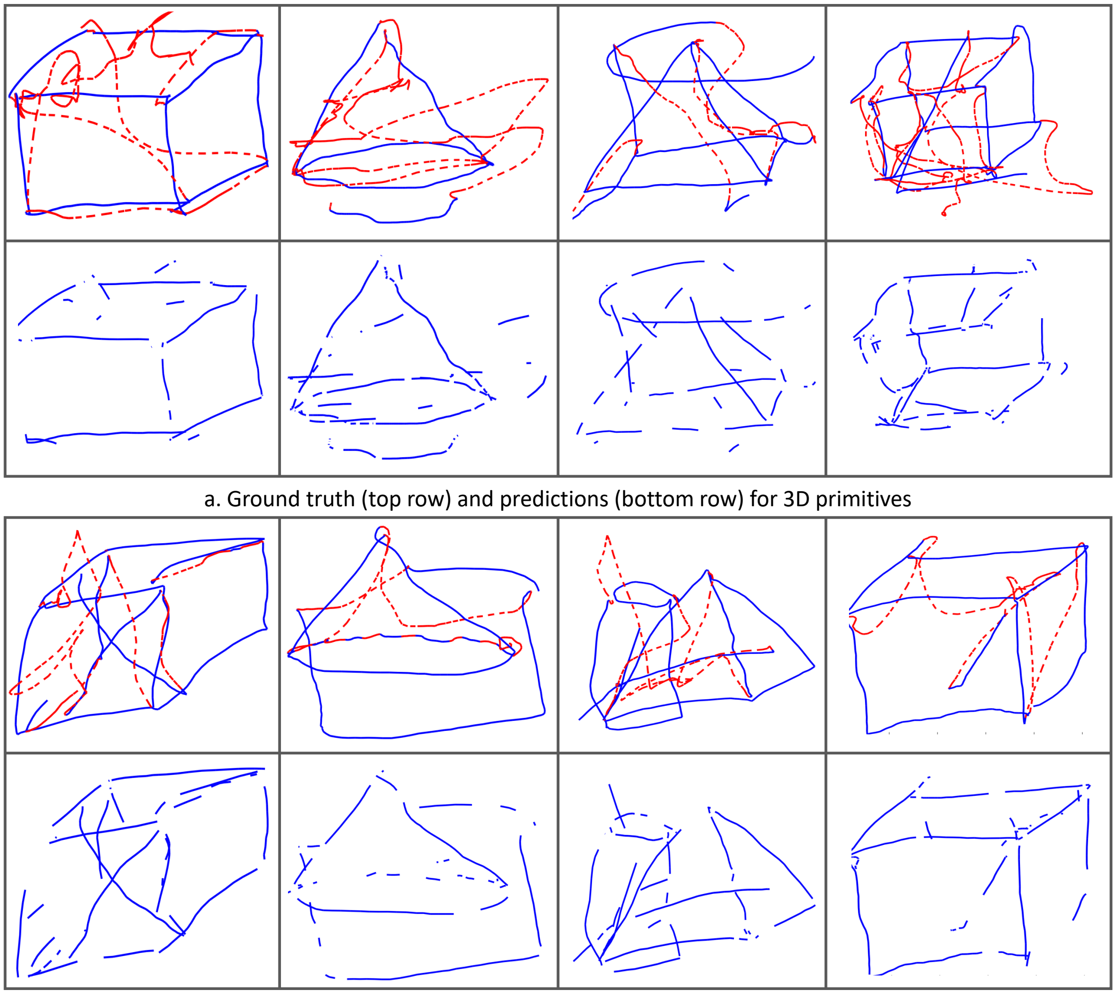


a. Ground truth (top row) and predictions (bottom row) for 3D primitives

b. Ground truth (top row) and predictions (bottom row) for 3D primitives

b. Ground truth (top row) and predictions (bottom row) for 3D primitives

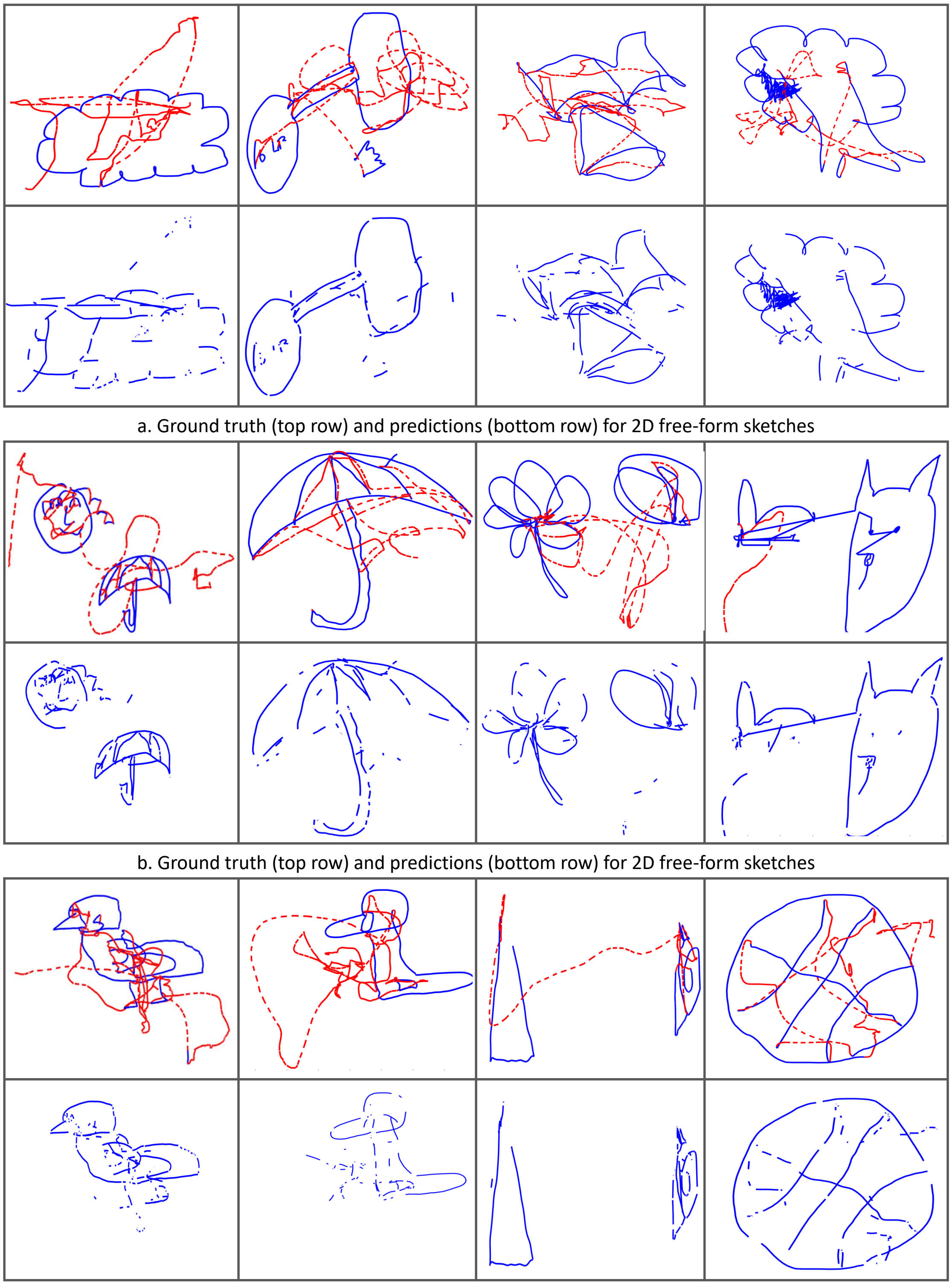
Figure A.3: Random Forest model predictions for 3D primitives & 2D free-form data recorded using GeoMagic Touch.



a. Ground truth (top row) and predictions (bottom row) for 3D primitives

b. Ground truth (top row) and predictions (bottom row) for 3D primitives

Figure A.4: Random Forest model predictions for 3D primitives data recorded using GeoMagic Touch.



a. Ground truth (top row) and predictions (bottom row) for 2D free-form sketches

b. Ground truth (top row) and predictions (bottom row) for 2D free-form sketches

c. Ground truth (top row) and predictions (bottom row) for 2D free-form sketches

Figure A.5: Random Forest model predictions for 2D free-form shapes recorded using GeoMagic Touch.

APPENDIX B

RAW DATA BASED CLASSIFICATION RANDOM FOREST PREDICTIONS

In this section, we show all random forest model prediction results all six shape categories sampled from the 50000 testing data points recorded across the 25 participants.

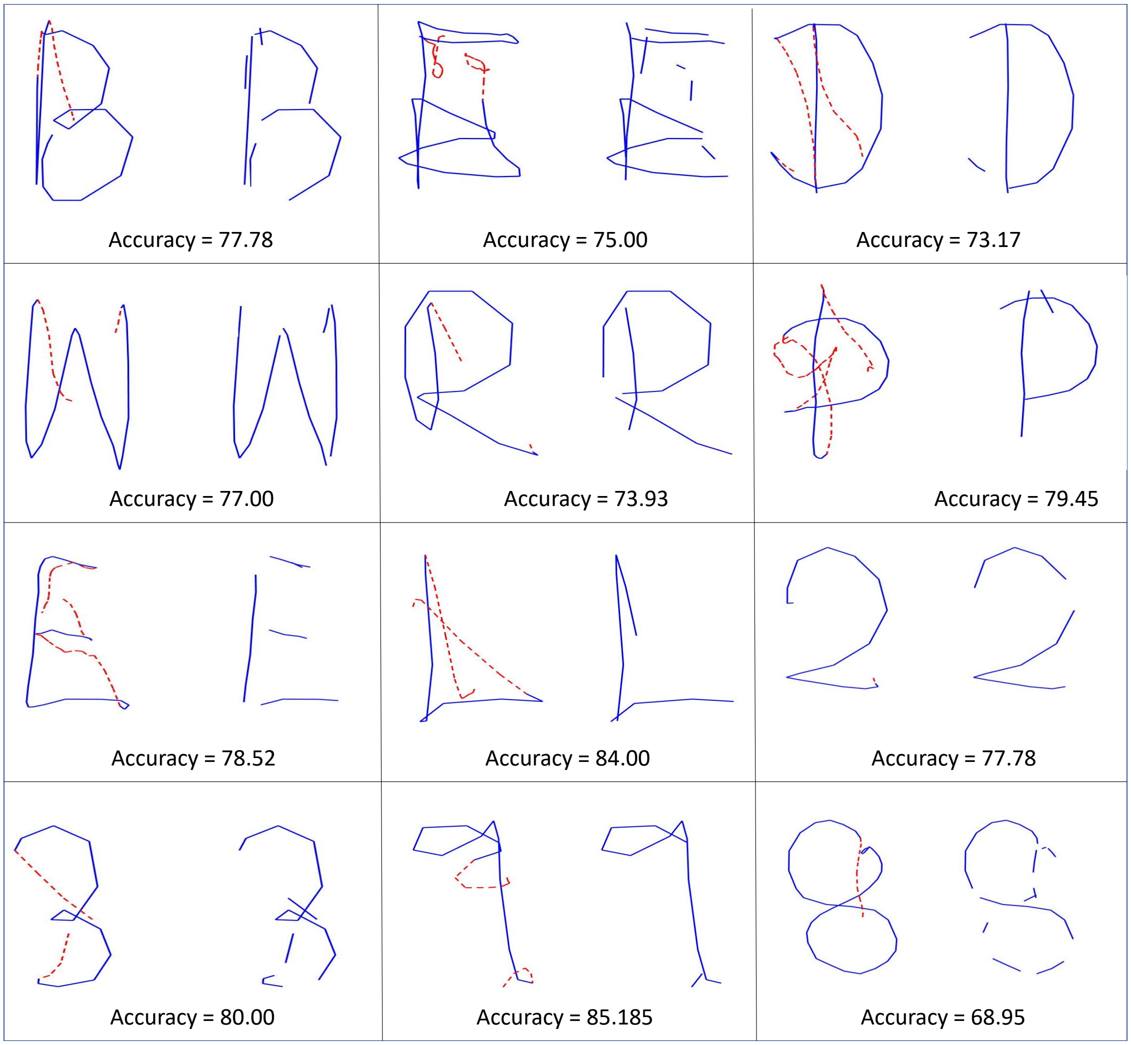


Figure B.1: Random Forest model predictions for shapes belonging to the alphanumerics category.

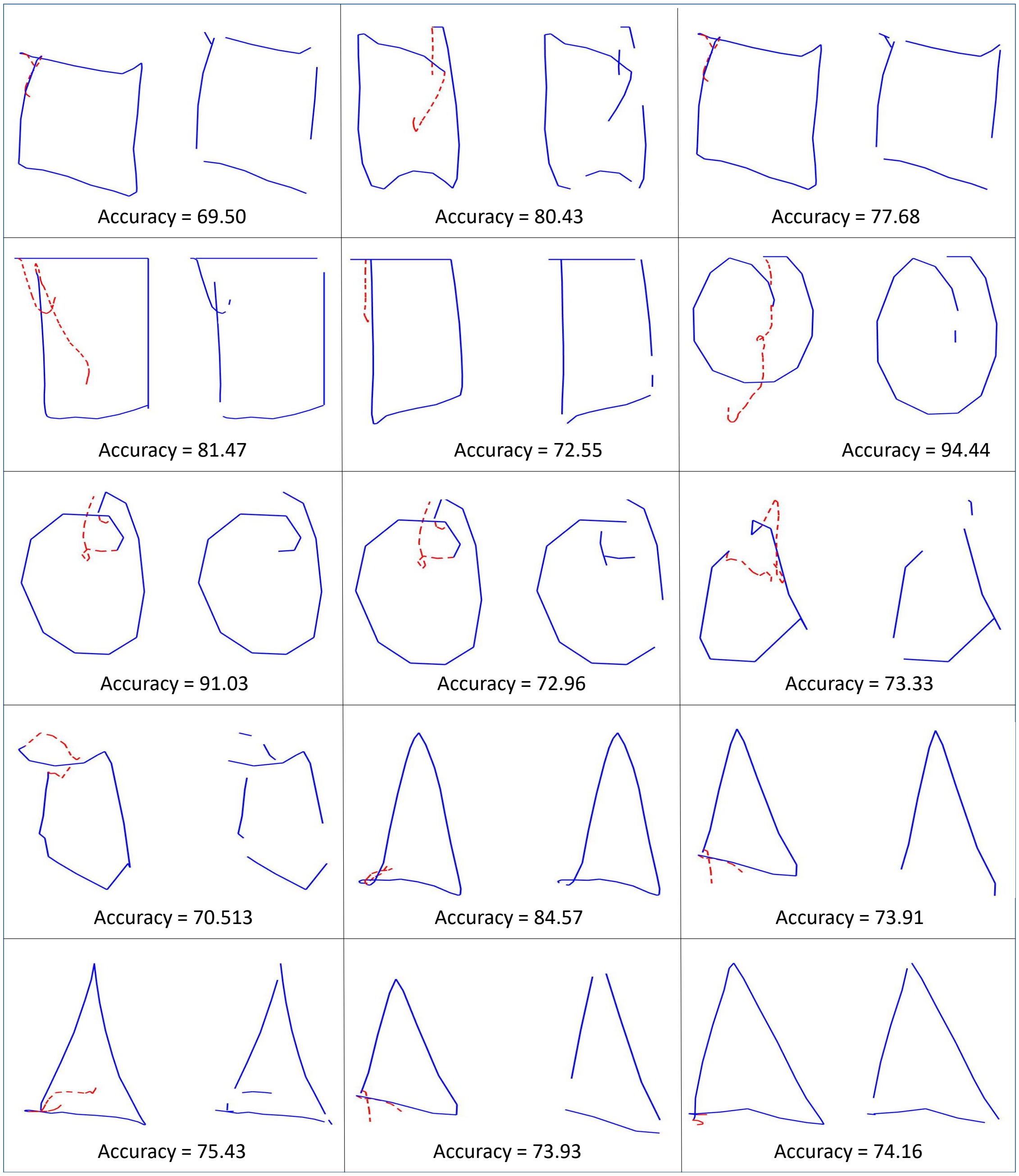


Figure B.2: Random Forest model predictions for shapes belonging to the 2D primitives category.

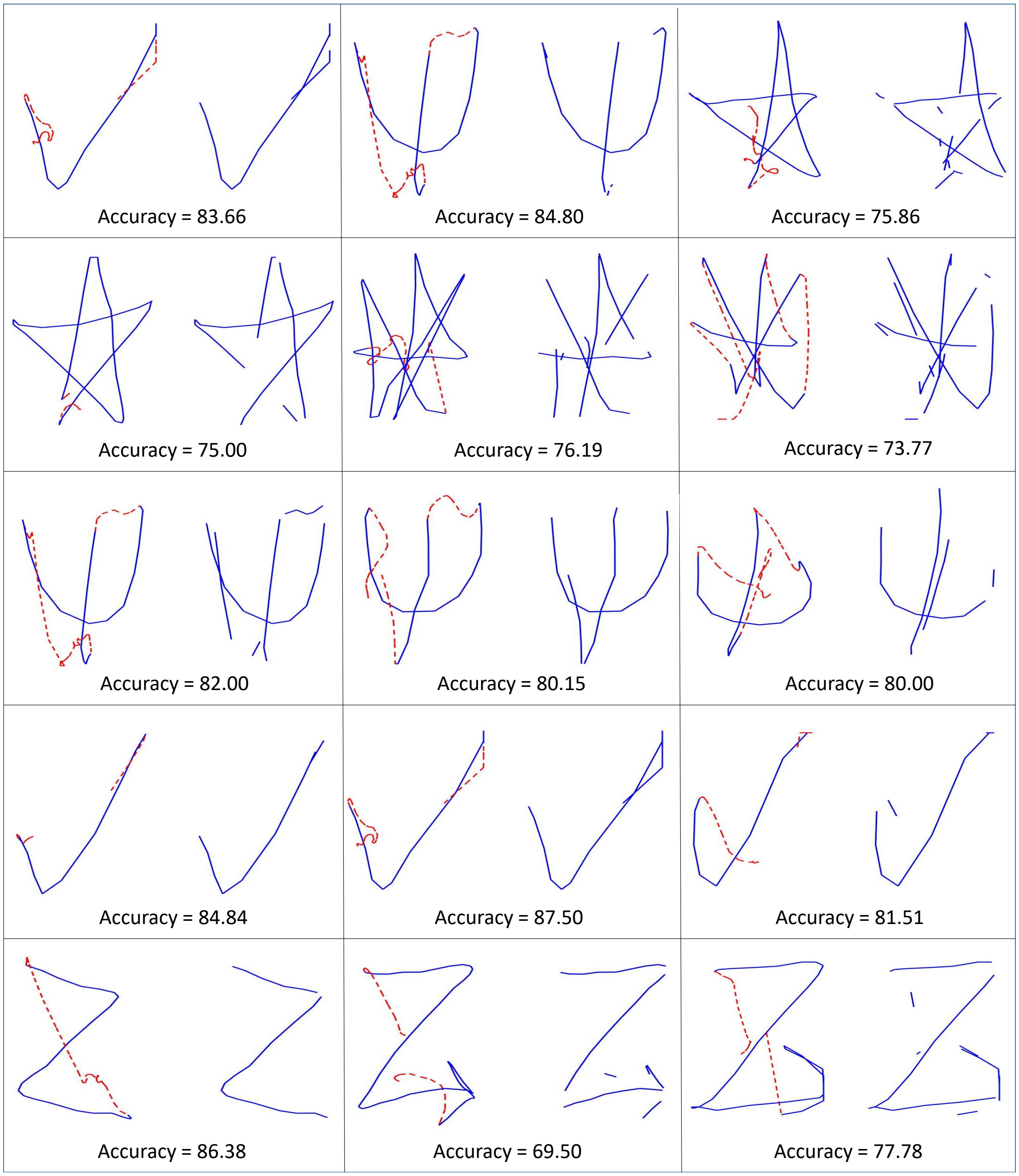


Figure B.3: Random Forest model predictions for shapes belonging to the gestures category.

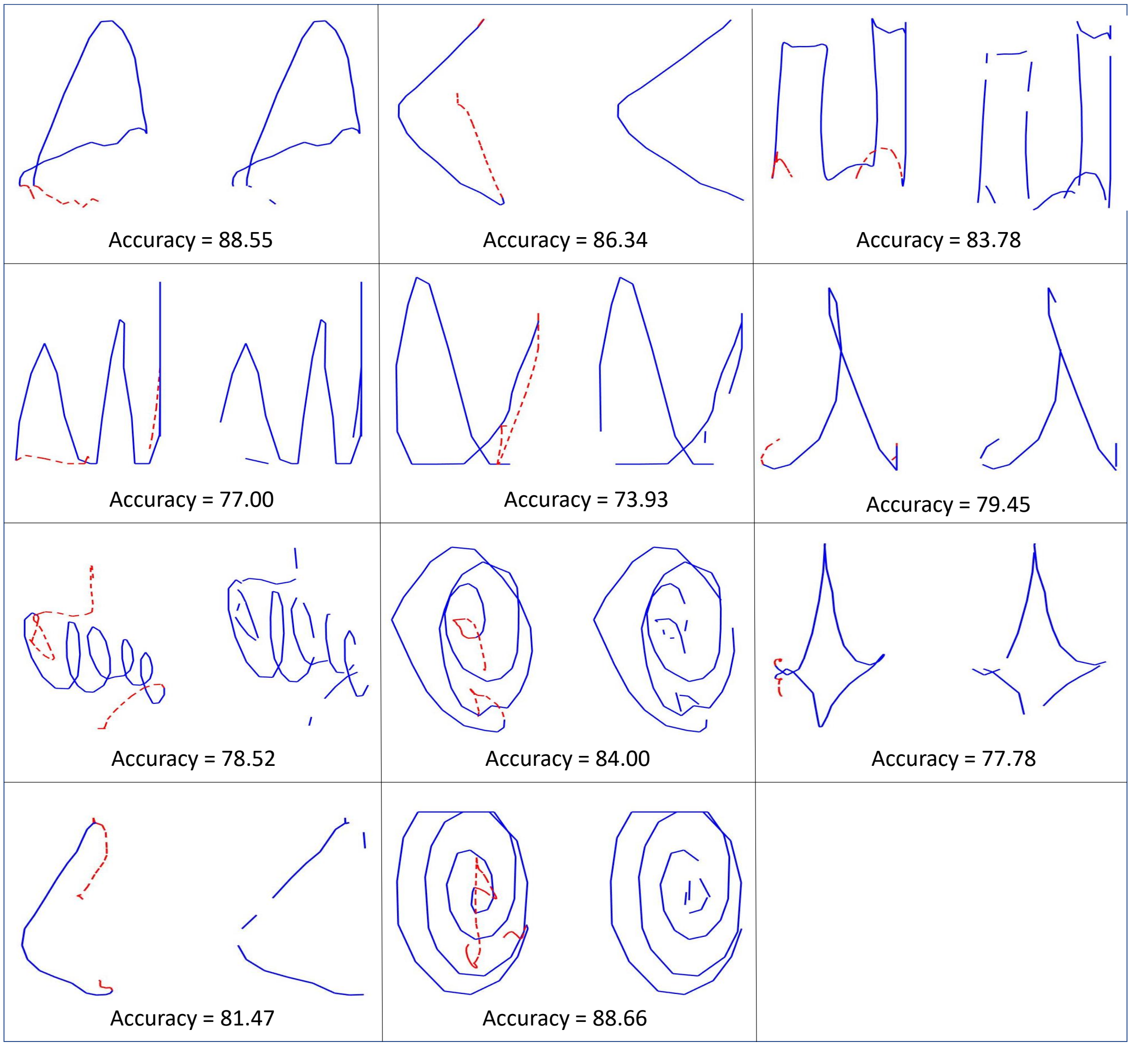


Figure B.4: Random Forest model predictions for special curves recorded using the data collection setup.