Int. J. Human-Computer Studies 73 (2015) 91–106

Contents lists available at ScienceDirect

Int. J. Human-Computer Studies

journal homepage: www.elsevier.com/locate/ijhcs

Gaze-based prediction of pen-based virtual interaction tasks ☆

Çağla Çığ *, Tevfik Metin Sezgin

Intelligent User Interfaces Lab, Department of Computer Engineering, Koç University, Istanbul 34450, Turkey

ARTICLE INFO

Article history:
Received 12 June 2014
Received in revised form
19 September 2014
Accepted 20 September 2014
Available online 28 September 2014

Keywords:
Sketch-based interaction
Multimodal interaction
Predictive interfaces
Gaze-based interfaces
Feature selection
Feature representation
Multimodal databases

ABSTRACT

In typical human–computer interaction, users convey their intentions through traditional input devices (e.g. keyboards, mice, joysticks) coupled with standard graphical user interface elements. Recently, pen-based interaction has emerged as a more intuitive alternative to these traditional means. However, existing pen-based systems are limited by the fact that they rely heavily on auxiliary mode switching mechanisms during interaction (e.g. hard or soft modifier keys, buttons, menus). In this paper, we describe how eye gaze movements that naturally occur during pen-based interaction can be used to reduce dependency on explicit mode selection mechanisms in pen-based systems. In particular, we show that a range of virtual manipulation commands, that would otherwise require auxiliary mode switching elements, can be issued with an 88% success rate with the aid of users' natural eye gaze behavior during pen-only interaction.

1. Introduction

Pen-based devices are gaining popularity. Pen-enabled smart phones and tablet computers have penetrated our lives to a great extent due to their mobility, ease of use and affordable prices. However, despite what their name suggests, pen-based devices are not purely pen-based (Plimmer, 2008).

For example, in pen-enabled smart phones, many actions force the user to put the pen aside and switch to multi-finger gestures (e.g. spread/pinch for zoom in/out, and swipe to navigate back/forward). These gestures require the simultaneous use of 2, 3 or even 4 fingers (Fig. 1a). The necessity of switching between pen and multi-touch input goes against the goal of seamless interaction in pen-based devices.

Even the state of the art devices and software specifically built for pen-based interaction lack purely pen-based interaction. For example, graphics tablets preferred mainly by digital artists such as Wacom Cintiq 24HD (Fig. 1b) are often referred to as “heaven on earth” by users. However, even with these high-end models many tasks are still accomplished via on-pen or on-tablet external buttons called “express keys”, “touch rings” and “radial menus”. These buttons allow the user to simulate keystrokes including letters, numbers and modifier keys (e.g. Shift, Alt and Control). To issue a virtual manipulation command (e.g. scroll), the user has to locate the correct button which interrupts the interaction flow, hence causing an overall disappointing experience.

Another example where we lose purely pen-based interaction is with tablet computers. In most pen-based applications, features are hidden in standard context/pop-up menus that are accessed via tapping and/or holding the pen on the tablet screen in various ways (Fig. 1c). In this case, the pen is used to trigger mouse clicks, which fits the traditional GUI/WIMP-based interaction paradigm, rather than that of a purely pen-based interaction (Fig. 1d).

These issues show that existing pen-based systems depend substantially on multi-finger gestures, context/pop-up menus and external buttons which goes against the philosophy of pen-based interfaces as a more intuitive interaction alternative. In this paper, we show that eye gaze movements that naturally accompany pen-based user interaction can be used to infer a user's task-related intentions and goals. The non-intrusive and transparent use of eye gaze information for task prediction brings us closer to the goal of purely pen-based interaction and reduces the reliance on multi-finger gestures, context/pop-up menus and external buttons.

Our approach consists of tracking eye gaze movements of the user during pen-based interaction, fusing the spatio-temporal information collected via gaze and sketch modalities in order to predict the current intention of the user. We use the term “intention” to refer specifically to the intention of the user to issue a virtual manipulation command. Virtual manipulation commands that we can currently predict are: *drag*, *maximize*, *minimize* and *scroll*. Additionally, we can distinguish whether the user intends to issue any of these virtual manipulation commands, or intends to sketch using our special-purpose task class called *free-form drawing*.

☆This paper has been recommended for acceptance by J. LaViola.

* Corresponding author. Tel.: +90 212 338 1540; fax: +90 212 338 1548.
E-mail addresses: ccig@ku.edu.tr (Ç. Çığ), mtsezgin@ku.edu.tr (T. Metin Sezgin).

http://dx.doi.org/10.1016/j.ijhcs.2014.09.005
1071-5819/

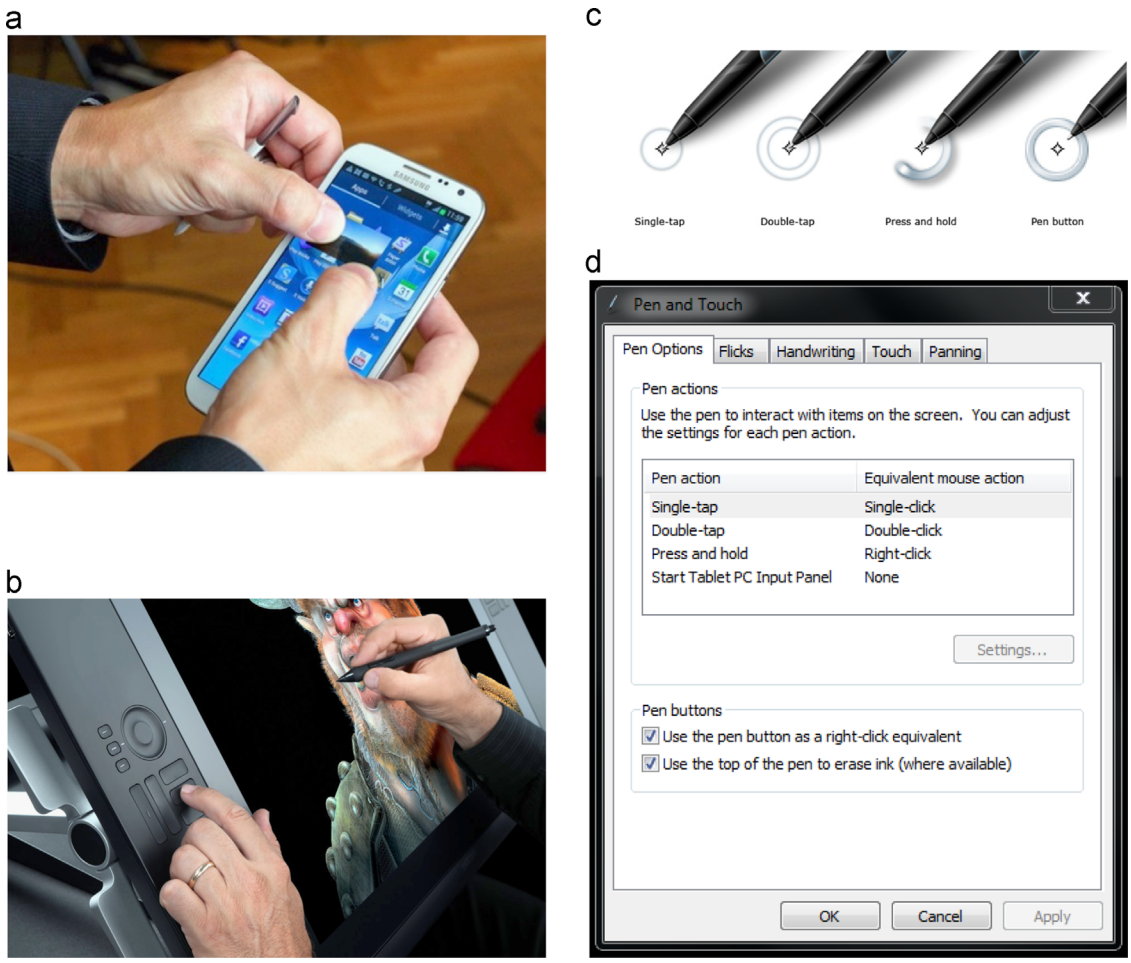

Fig. 1. Some examples to pen-based devices with interaction paradigms that are not purely pen-based. Gaze-based prediction of virtual interaction tasks in pen-based interaction is a step towards systems that require fewer mode changes (Li et al., 2005). (a) Switching between pen and multi-touch input for object manipulation (e.g. image resizing) in pen-based smart phones. (b) On-pen or on-tablet external buttons in pen-based graphics tablets. (c) Various tapping and/or holding techniques to access context/pop-up menus in pen-based tablet computers. (d) The pen is used to emulate a mouse in pen-based tablet computers.

Our overall approach to gaze-based prediction of virtual interaction tasks is depicted in Fig. 2. The left part of the diagram shows how we build our system whereas the right part shows how our system performs predictions. After we build our system, it can be used to infer user's task-related intentions and goals in an event-driven manner where each pen marking triggers prediction.

Briefly, our system is built as follows: Initially we collect sketch and gaze data during a number of pen-based interaction tasks and build a multimodal database. We then extract novel gaze-based features from this database and train a task prediction model using supervised machine learning techniques. These steps are executed only once. Then, our system is ready for prediction. When the user performs a pen action (demarcated by a pen-down and a pen-up event), the synchronized pen trajectory and eye gaze information is used to predict the user's intended virtual task. Predictions are carried out by the previously trained model and the features extracted from the corresponding sketch–gaze data of the user. Detailed description and discussion of our approach can be found in the following sections.

We have three main contributions. First, we present a carefully compiled multimodal dataset that consists of eye gaze and pen input collected from participants completing various virtual interaction tasks. Second, for predicting user intention through gaze, we propose a novel gaze-based feature representation based on human vision, and behavioral studies. Third, we introduce a novel gaze-based task prediction system that uses this feature representation. These features are neither subject- nor interface-specific, and perform better than commonly utilized and well-established sketch recognition feature representations in the literature. We evaluate our system based on several aspects, including the prediction accuracy and scale-invariance. In addition, we run feature selection tests to evaluate the relevance and redundancy of the feature representations. Our prediction system opens the way for more natural user interface paradigms where the role of the computer in supporting interaction is to "interpret user actions and [do] what it deems appropriate" (Nielsen, 1993). It is widely accepted that intelligent mode selection mechanisms that provide low cost access to different interface operations will dominate new user interface paradigms (Negulescu et al., 2010).

Section 2 gives an outline of the state-of-the-art gaze-based interfaces in a categorical manner with relevant examples for each category. Our approach consists of three major parts: data collection, feature extraction and intention prediction. These parts are detailed in Sections 3–5, respectively. Section 6 concludes with a discussion of our work and a summary of future directions.

2. Related work

We have presented a novel gaze-based interface for predicting virtual manipulation commands during pen-based interaction. State-of-the-art gaze-based interfaces fall under two main categories: *command interfaces* and *non-command interfaces*.

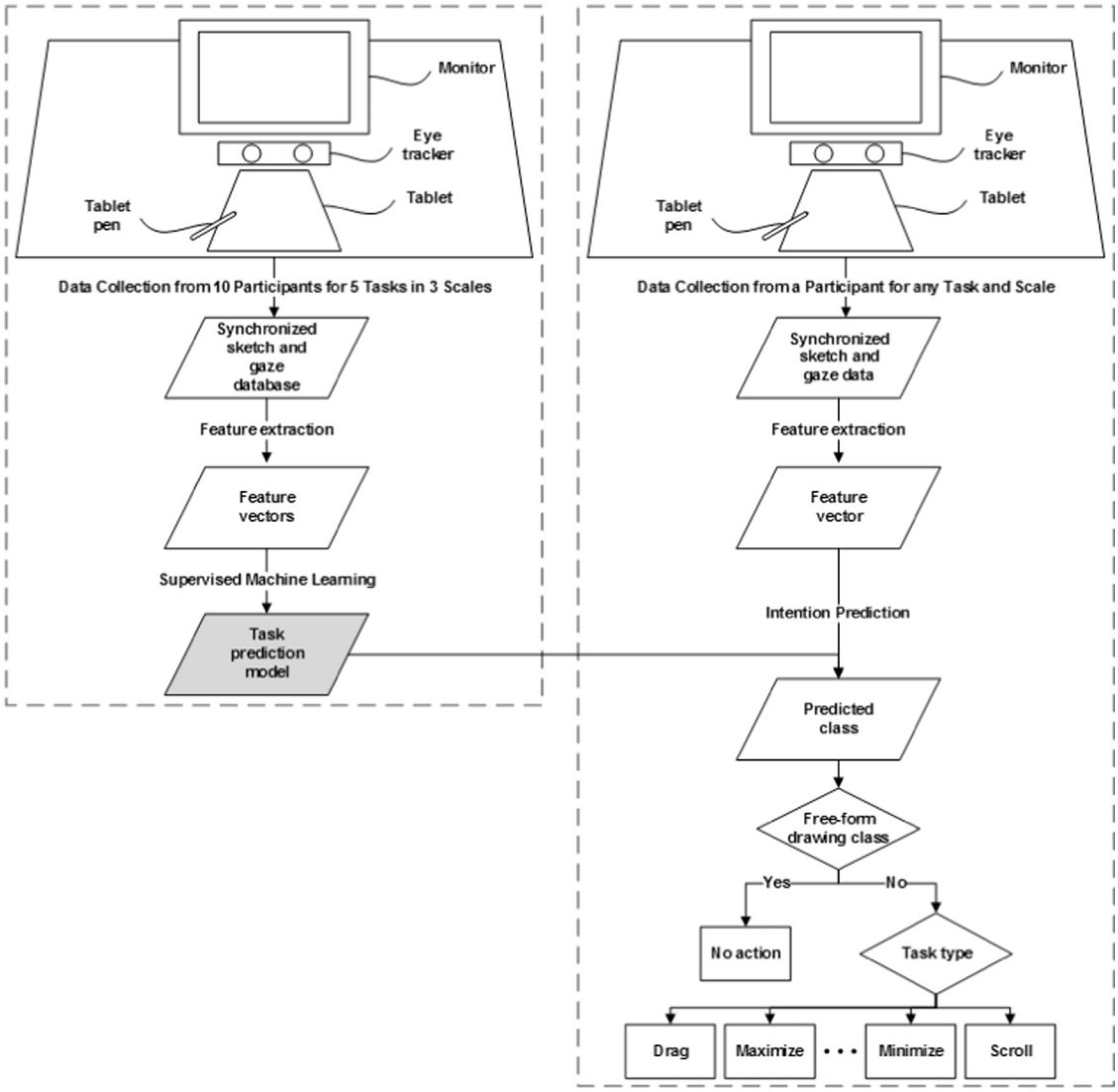

Fig. 2. Flow diagram visualizing our overall approach to gaze-based prediction of virtual interaction tasks. The figure on the left illustrates how we build our system, and the one on the right shows how our system works in practice.

Command interfaces are based on the eye-mind hypothesis, in which, intentional eye movements are associated with interface actions (Bednarik et al., 2012). In other words, in command interfaces, gaze is employed as an explicit pointing device. This requires the gaze to be used for manipulation in addition to its natural purpose, visual perception. This approach forces the user to be aware of the role of the eye gaze and therefore results in high cognitive workload (Bader et al., 2009). Zhai et al. argue that "other than for disabled users, who have no alternative, using eye gaze for practical pointing does not appear to be very promising" (Zhai et al., 1999). Kumar et al. agree that "overloading the visual channel for a motor control task is undesirable" (Kumar et al., 2007). In line with these arguments, our work avoids forcing the user to consciously adopt unnatural gaze behavior for interaction purposes and instead uses gaze movements that naturally accompany manipulation tasks for prediction.

In non-command interfaces, the computer system passively and continuously observes the user in real-time and provides appropriate responses. In order to provide satisfying and natural responses, the computer system must be able to infer user's intentions from his/her spontaneous natural behaviors. An intention can be, for instance, moving a window, scrolling a piece of text or maximizing an image (Bednarik et al., 2012). Studies (Bader et al., 2009; Iqbal and Bailey, 2004; Yu and Ballard, 2002a; Hayhoe and Ballard, 2005; Bulling et al., 2011; Campbell and Maglio, 2001) provide qualitative observations and quantitative evidence suggesting that well-structured tasks have unique eye movement signatures. However, the majority of the related work on non-command interfaces focuses solely on post-hoc analysis of eye movements collected during natural interaction. Only a few researchers have made considerable attempts at interpreting user behavior for online task prediction. Therefore, non-command interfaces can be grouped under two subcategories: *task analyzers* and *task predictors*. Both categories can be further divided as *daily task analyzer/predictors* and *virtual task analyzer/predictors* depending on the nature of tasks taken into consideration. Daily tasks such as sandwich making and stapling a letter are ordinary activities in everyday settings (Land and Hayhoe, 2001; Yi and Ballard, 2009; Yu and Ballard, 2002a, 2002b; Fathi et al., 2012) whereas virtual tasks such as reading an electronic document or manipulating a virtual object involve the use of a computer system (Alamargot et al., 2006; Gesierich et al., 2008; Campbell and Maglio, 2001; Bader et al., 2009; Bednarik et al., 2012). We focus on pen-based tablet devices; therefore, we are interested in analyzing and predicting the range of virtual interaction tasks commonly performed on tablet devices. However, for completeness sake, our literature review covers daily tasks as well. In the following

subsections we provide a review of the related work that fall under each category. A more comprehensive summary of related work can be found in Appendix A.

2.1. Daily task analyzers

Daily task analyzers focus on analyzing various characteristics of eye movements while users perform daily tasks. Land and Hayhoe (2001) investigate the relationships between eye and hand movements in food preparation tasks such as brewing tea and fixing a sandwich. Their results are largely composed of plots displaying how body, eye and hand movements change in time and the authors point out that the control of eye movements is primarily directed at the ongoing motor actions. Yi and Ballard (2009) also focus on the sandwich making task; however, they take a more probabilistic approach. The authors manually segment the task into subtasks such as locating the bread, spreading jelly on the bread etc., and then use a dynamic Bayesian network (DBN) to model the task. However, the authors do not assess the goodness of their predictions and only provide graphs visualizing the real and inferred timings of the subtasks. Lastly, Hayhoe and Ballard (2005) present a review of approaches that analyze eye movements in everyday visually guided behaviors, however these do not interpret user behavior for online task prediction.

2.2. Virtual task analyzers

Virtual task analyzers focus on analyzing various characteristics of eye movements for tasks that involve the use of a computer system. Iqbal and Bailey (2004) study eye gaze patterns in four different tasks: reading comprehension, mathematical reasoning, search and object manipulation. The authors segment the virtual interaction area into interface-specific areas of interest (AOI) and qualitatively inspect the relationship between amount of time spent on each AOI and task type. They show that the percentage of time spent on each AOI varies across task categories. However, they do not provide statistical analysis or a mechanism for prediction. Alamargot et al. (2006) provide a detailed description of the eye movement characteristics recorded during reading and writing on a digital tablet. Gesierich et al. (2008) observe proactive (i.e. anticipatory) eye movements in both action execution and action observation during a two-user virtual block stacking task. Our work differs from the listed virtual task analyzers in its successful attempt at interpreting user behavior for online task prediction.

2.3. Daily task predictors

Yu and Ballard (2002a), Yu and Ballard (2002b) use Hidden Markov Models (HMM) to discriminate between the tasks of unscrewing a jar, stapling a letter and pouring water. In addition to features related to the eye, head and hand movements, the authors also employ features related to scene objects being fixated by the user during task execution. This substantially simplifies task prediction as it practically reduces to discriminating between a jar, a stapler and a carafe. Similarly, in more recent work, Fathi et al. (2012) use Support Vector Machines to discriminate numerous subtasks of a meal preparation task. The authors extract gaze-related features and features from the fixated scene objects that mainly describe their key visual properties. More specifically, Yu and Ballard (2002a), Yu and Ballard (2002b) demonstrate that the user most probably has an intention to staple a letter and not unscrew a jar or pour water if the object of focus is a stapler. However, it is certainly less clear whether the user intends to drag, maximize, minimize, scroll or sketch on a virtual object; for instance, an image, even if the object of focus is the image itself. The ultimate aim of task predictors is not predicting the current task in a certain context but doing so in a context-independent way. All pieces of work focus on daily task prediction and utilize context information. This is feasible in the context of daily tasks; however, not as much so in the case of virtual tasks since the user might be performing a variety of different tasks while focusing on the same virtual object.

Bulling et al. (2009, 2011) present two closely related works. The authors focus on classifying tasks in three domains: reading, office activities and cognitive psychology (more specifically, visual memory recall). The domain most related to our work is office activities consisting of copying, reading, writing, watching a video and browsing the Web. They report an average precision score of 76.1% using SVMs, which is comparably higher than scores reported by related works that focus on gaze-based task prediction. A more recent closely related work is by Ogaki et al. (2012). They study the same indoor office activities as Bulling et al. (2009, 2011), Bulling et al. (2009) and demonstrate that coupling eye movements with ego-motion leads to better task classification performance. Their work resembles our work in its use of multiple modalities, however the features used in these studies depend heavily on preset templates to track repetitive patterns of eye movements and on constants for defining threshold levels, sliding window sizes, etc. On the contrary, the highly generic features of our current work eliminate the need for such possibly subject- and interface-specific preprocessing steps. Moreover, they report a comparably low average precision score of only 57%.

2.4. Virtual task predictors

Among the earliest examples of virtual task predictors is work by Campbell and Maglio (2001). They use a wide range of eye movement patterns in order to classify reading, skimming and scanning tasks.

This was followed to a great extent by studies concentrating on intention prediction, i.e. predicting whether the user wants to interact with the system or not during natural interaction. For instance, Bader et al. (2009) use a probabilistic model to predict whether the user intends to select a virtual object or not with 80.7% average accuracy. Similarly, Bednarik et al. (2012) use SVMs to predict whether the user intends to issue a command or not with 76% average accuracy. Both prediction tasks are examples of binary classification, which indicates that in both cases, baseline accuracy score corresponding to the random classifier is 50%. This cannot be compared to our case where the baseline accuracy score is merely 20%. Hence, reported accuracy scores and classifier gains (defined as the measurement of improvement over the random classifier) should be interpreted in consideration of this fact.

To the best of our knowledge, only a few studies exist that take intention prediction one step further and attempt multi-class intention prediction of virtual tasks. The first notable example is by Courtemanche et al. (2011) who claim their approach to activity recognition to be the first one to incorporate eye movements. This work utilizes eye movements discretized in terms of interface-specific AOIs in addition to keystrokes and mouse clicks input by the user during interaction. They use HMMs to predict which of the three Google Analytics tasks (i.e. evaluating trends in a certain week, evaluating new visits and evaluating overall traffic) the user is currently performing with 51.3% average accuracy. The second example is recent work by Steichen et al. (2013). Their domain is information visualization with graphs including bar graphs and radar graphs. Similarly, they rely on interface- and graph-specific AOIs for feature extraction and Logistic Regression to predict which of the five information visualization tasks (retrieve value, filter, compute derived value, find extremum and sort) the user is currently performing with 63.32% average accuracy.

The superiority of our work over these two studies is threefold. First, both of these studies are interface-dependent since they analyze eye movements with respect to predefined AOIs. In contrast, our work avoids possibly subject- and interface-specific preprocessing steps common in gaze-based systems. Second, possible application areas of both of these studies are highly specific and limited since the corresponding task prediction models focus on Google Analytics tasks and graph-based information visualization tasks, respectively. On the contrary, our work can be applied in all areas that utilize basic interaction tasks like dragging, resizing and scrolling. Accordingly, the application areas can range from simple interfaces to more complicated document or image editing software. Third, our prediction system is comparably more accurate, which makes it a better candidate for practical use.

3. Multimodal data collection

We interpret pen and eye gaze input within a machine learning framework. This primarily requires large amounts of data for training classifiers. We collect data in a controlled setup where the users are asked to carry out a number of pen-based virtual interaction tasks.

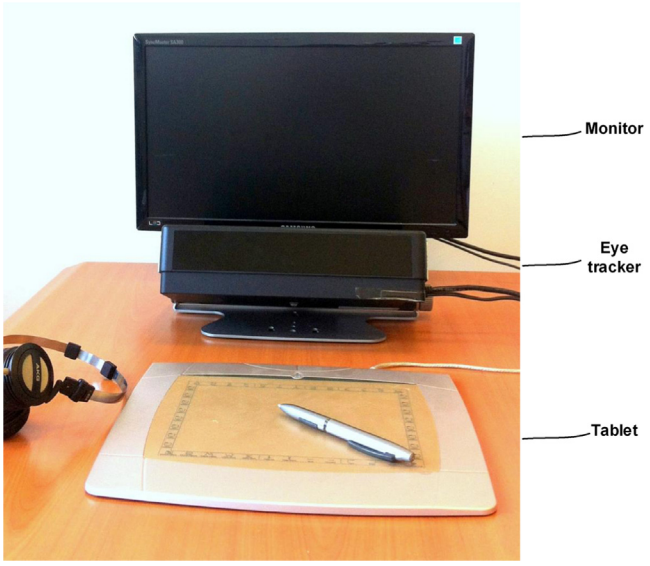

Fig. 3. Physical setup for multimodal data collection. Input and display are separated resulting in an indirect input configuration (Forlines and Balakrishnan, 2008).

3.1. Physical setup

To create a database composed of synchronized sketch and gaze data, we use a tablet and a Tobii X120 stand-alone eye tracker for the sketch and gaze modalities, respectively. The eye tracker needs to be calibrated once for each user. The calibration step is brief, and it is posed as an "attention test" to conceal any hints of eye tracking from the user. Tobii X120 operates with a data rate of 120 Hz, tracking accuracy of 0.5° and drift of less than 0.3°. The tracker allows free head movement inside a virtual box with dimensions $30 \times 22 \times 30$ cm.

The physical setup for data collection is depicted in Fig. 3. Note that the drawing surface and the display are separated. In particular, the drawing surface (i.e. a tablet) is placed below the eye tracker and the eye tracker is placed below a monitor. This physical configuration allows us to collect pen input given the technical limitations of the Tobii X120 eye tracker. More specifically, the general setup guidelines for the eye tracker require placing it below the interaction screen. However, placing the eye tracker below the tablet inevitably leads to user's arm blocking the eye tracker's field of operation. To overcome this problem, the drawing surface and the display are separated in our setup. To facilitate hand–eye coordination during interaction, we render a pen-shaped visual cursor on the display indicating the position of the user's pen on the tablet.

3.2. Data collection tasks

Our data collection process is designed to include frequently employed pen-based virtual interaction tasks. These tasks, depicted in Fig. 4, are: *drag*, *maximize*, *minimize*, *scroll* and *free-form drawing*. Typical pen-based interaction consists of *stylized* and *non-stylized pen inputs*. Stylized pen inputs consist of symbols and gestures which have characteristic visual appearances (Fig. 5). Hence, they can be classified with conventional image-based recognition algorithms. On the other hand, non-stylized pen inputs lack a characteristic visual appearance, and appearance alone does not carry sufficient information for classification purposes. Therefore, in order to test out our system's prediction

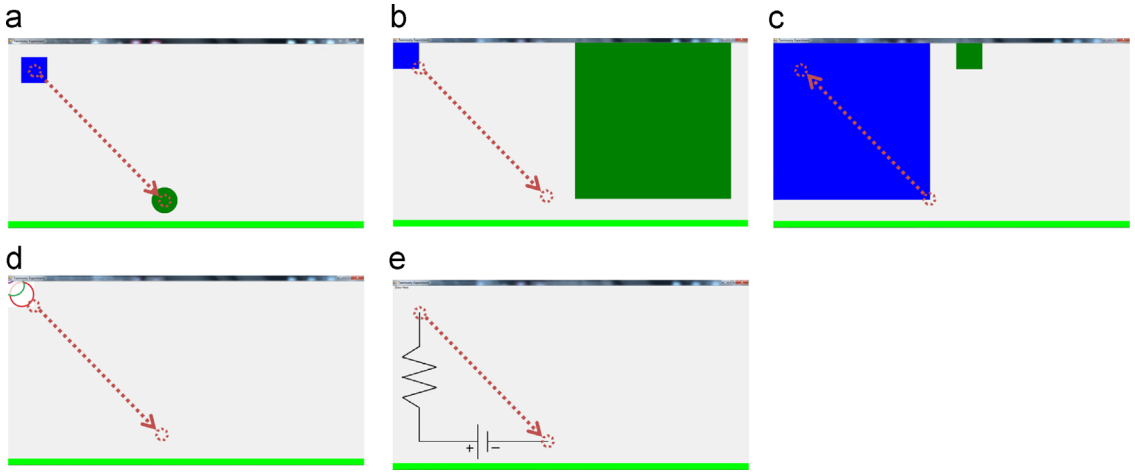

Fig. 4. Pen-based virtual interaction tasks included in our research. Starting and ending regions of desired pen motion in each task are visualized with dotted circles. In the rest of the article, the center points of these regions will be referred to as *anchor points*. Direction of the desired pen motion in each task is visualized with a dotted arrow connecting the starting and ending regions. It is important to note that the dotted visualizations only serve as a reference within this paper and they are not shown to the user during data collection. (a) Drag. (b) Maximize. (c) Minimize. (d) Scroll. (e) Free-Form Drawing. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

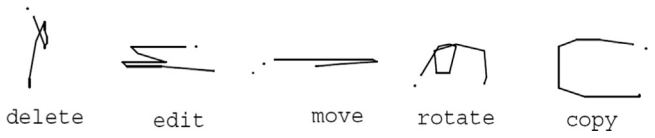

Fig. 5. Common editing gestures (Rubine, 1991) serve as examples of stylized pen input. Each gesture has an easily distinguishable characteristic visual appearance.

Table 1
Instructions given to the users for each task during data collection.

Task	Instruction
Drag	Drag the blue square onto the center of the green circle
Maximize	Increase the size of the blue square to match the size of the green square
Minimize	Decrease the size of the blue square to match the size of the green square
Scroll	Pull the chain until the color of the last link is clearly visible
Free-form drawing	Connect the battery and the resistor with a wire

power in a more challenging setting, we selected tasks that yield non-stylized pen input. In particular, for each task the stylus has an approximate starting point and an approximate ending point. In order to complete each task, the user needs to make a movement that starts near the starting point and ends near the ending point. Pen input corresponding to these tasks do not have characteristic visual appearances and do not lend themselves well to conventional image-based recognition algorithms. Instructions given to the users for each task are summarized in Table 1.[1]

In addition to being frequently employed, these tasks also have the following properties in common:

- Each task can be carried out using a tablet and a stylus.
- Each task necessitates continued visual attention for planning and guiding the hand/eye movements. Thus during each task, user's eyes are expected to remain on the display device.
- Tasks last roughly the same amount of time.

The *Free-form drawing* task differs from the remaining tasks in a special way. Unlike the other pen-based virtual interaction tasks, this task is included in our study in an attempt to model and avoid unsolicited task activation. More specifically, if our prediction system is to be employed in a proactive user interface, the ability to distinguish between the intention to sketch and the intention to interact becomes vital. Accordingly, the *free-form drawing* task is included in our study to distinguish pen movements that are intended to activate the proactive user interface for task execution. Otherwise, the user would find that all pen movements (intended or not) activate a new task execution.

3.3. Data collection interface

To collect multimodal data for the mentioned tasks, we designed and implemented a custom application. We collected sketch data using the Microsoft Managed INK Collection API (Pen API) and INK Data Management API (Ink API). These APIs capture pen coordinates online, and save digital ink packets captured between pen-down and pen-up events as strokes. We collected gaze data using the Tobii Analytics Software Development Kit (SDK). The collected gaze data is composed of gaze points, each represented as an array of tuples consisting of local UNIX timestamp, remote UNIX timestamp, validity code, horizontal location and vertical location information sampled at 120 Hz. Validity code, horizontal location and vertical location information are obtained for the left and right eyes individually.

Our user interface has the following properties:

- Sketch and gaze data are collected in a time-synchronized fashion.
- A gaze tracking status bar visualizes whether the eye tracker is able to find both eyes. Users can monitor and adjust their posture based on the gaze tracking status bar. The bar stays green as long as the eye tracker is functioning properly, but turns red if the eye tracker loses the eyes. Gaze data packets collected while the status bar is red are marked as *invalid* by the eye tracker. The status bar is disguised under the name of a *posture check indicator* in an attempt to avoid any hints of eye tracking.
- At the beginning of each task, prerecorded non-distracting (in terms of avoiding unsolicited gaze behavior) audio instructions are delivered via headphones.
- After the completion of each task, the percentage of *valid* gaze data is calculated to assure that at least 80% of the collected gaze data is *valid*. In cases where fewer than 80% of the gaze packets are valid, the current task is automatically repeated and the user is warned via an audio message instructing him/her to assume a correct posture and maintain an appropriate distance to the monitor.

When users execute a task, positions of the pen-down and pen-up events respectively define the starting and ending points of the task. To insure that starting and ending points of a task do not act as confounding variables in our data collection process, the tasks were designed to have coincident starting/ending points (Fig. 4).

3.4. Database

We refer to each run of a certain task at a certain scale as *a task instance*. Our multimodal database consists of 1500 task instances collected from 10 participants (6 males, 4 females) over 10 randomized repeats of 5 tasks across 3 scales. The participants were recruited from undergraduate and graduate students of Koç University's Faculty of Engineering on a voluntary basis.

Multimodal data was collected across three different scales to test our system in terms of scale-invariance. The scale variable determines the length of the path connecting the two anchor points present in each task (Fig. 4). These three scales will be referred to as *large*, *medium* and *small*, respectively. Lengths of the paths corresponding to each scale were set in light of facts about human vision. The spatial field of vision is functionally divided into three regions, foveal, para-foveal and peripheral. As summarized in Table 2, each region has distinct characteristics with respect to acuity limitations; therefore, lengths of the paths were

Table 2
Different regions of human spatial field of vision (Rayner, 2009).

Type of region	Foveal	Para-foveal	Peripheral
Limit (in degrees)	<2	2–10	>10
Limit (in cm)[a]	<2.44	2.44–12.25	>12.25

[a] Calculated based on acuity limit of each type of region in degrees and distance of the user to the monitor which is 70 cm in our setup.

[1] Note that the instructions for the *drag*, *maximize* and *minimize* tasks contain color information which will not show in a B/W copy of Fig. 4. For these tasks, the object to be manipulated (dragged/maximized/minimized) is the one on the left side of each screen.

set to 21 cm, 10.5 cm and 5.25 cm for the *large*, *medium* and *small* scales, respectively.

At the beginning of each data collection process, users were presented with 10 practice runs consisting of one run in *large* scale and one run in *small* scale for each of the 5 tasks.

Sketch data timestamps of two task instances were missing; thus, those instances were omitted from the database. In addition, invalid gaze data was filtered out using the validity codes.[2]

4. Novel gaze-based feature representation

Our system utilizes only two kinds of features for gaze-based task prediction: *Instantaneous Distance Between Sketch and Gaze Positions* and *Within-Cluster Variance of Gaze Positions*. The strength of these features stems from the fact that they eliminate the need for possibly subject- and interface-specific preprocessing steps common in gaze-based systems. Some examples of these common error-prone preprocessing steps include segmentation of gaze data into fixations and saccades and manual specification of regions of interest. Below, we describe each feature in detail, as well as the rationale behind how they are expected to aid task identification.

4.1. Feature 1: instantaneous distance between sketch and gaze positions

Let $G_t\langle x, y\rangle$ be the x and y positions of the gaze on the screen at time t during the execution of a particular task. Let $P_t\langle x, y\rangle$ represent the position of the tip of the stylus at time t. We argue that the distance between these points $D_t = |G_t - P_t|$ evolves in a strongly task-dependent fashion throughout the completion of a task instance. In other words, distance curves D_t computed for task instances of the same type have similar rise/fall characteristics, while those of different task types have quite different profiles. Unfortunately, even for the same task, the distance curves will evolve at different rates, hence they will not be identical. Assuming that we could compute representative characteristic curves for all task types, we could then compare the distance curve of an unknown task instance to these characteristic curves, and use the degree of matching as a useful feature for task identification. Below we describe the rationale behind the instantaneous distance feature, and suggest a method for computing task-specific characteristic curves.

4.1.1. The rationale

Hand–eye coordination behavior inherent in virtual interaction tasks changes over the course of a task instance as a function of changes in user sub-tasks (Hayhoe and Ballard, 2005; Fathi et al., 2012; Johansson et al., 2001; Ballard et al., 1992). The multiple steps of each task can be thought of as consecutive sub-tasks and each sub-task entails a different type of hand–eye coordination behavior. The rationale behind the first feature of our novel gaze-based feature representation is based on this observation and attempts to capture the goal-dependent dynamic aspects of human hand–eye coordination behavior through the evolution of the distance between instantaneous gaze and sketch positions calculated over a task instance.

Consider the task in Fig. 4a. In a typical instance of this task, the user is instructed to drag a source object (the blue square) onto a target object (the green circle). The sub-tasks of this task are

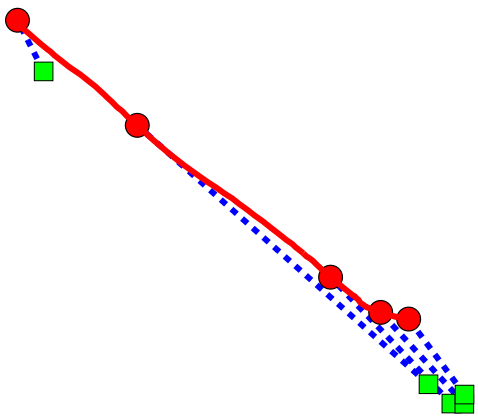

Fig. 6. Visualization of the user's sketch data (solid line) along with a number of sketch (circles) and gaze (squares) data samples. Dotted lines connect the instantaneous sketch and gaze sample pairs.

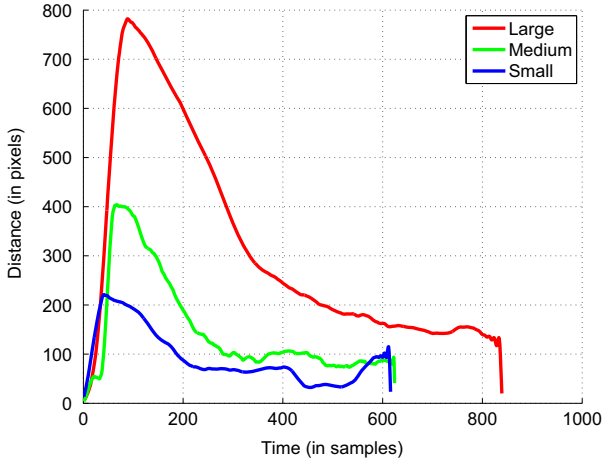

Fig. 7. Visualization of the changes in the value of sketch–gaze distance feature as a function of time.

(1) positioning the pen on the source object, (2) determining the position of the target object and (3) dragging the source object towards the target object. We argue that the dynamic aspects of human hand–eye coordination behavior are reflected in the distance values between instantaneous gaze and sketch positions calculated over time. Figs. 6 and 7 generated from the same sample task instance support our argument. Fig. 6 gives a visualization of the user's sketch data along with a number of sketch and gaze data samples. Sketch and gaze data points collected at identical time instances are connected with dotted lines. The length of a connection line denotes the value of the sketch–gaze distance feature for the corresponding instance. Fig. 7 demonstrates how the value of this feature changes over time. In this figure, the sketch–gaze distance feature is plotted for the same user and same task, over three different scales. Peaks of the plots could conceivably mark the second sub-task during which the user, after having positioned the pen on the source object, is now gazing at the target object. Note that sketch–gaze distance feature expresses similar characteristics across different scales; thus our approach and our novel feature can be generalized and applied to data collected across different scales.

Inspection of the sketch–gaze distance curves for the *drag* task reveals that the rapid rise and gradual decline behavior is typical of all instances of the *drag* task. Similarly, the distance curves for the other tasks also display task-specific characteristic rise and fall behaviors. We compute distinct sketch–gaze distance curves for

[2] Validity code information is an estimate of how certain the eye tracker is able to correctly identify both eyes. Validity codes can take on a set of specific values. The Tobii SDK Developer's Guide recommends all samples with validity codes 2 or higher to be discarded.

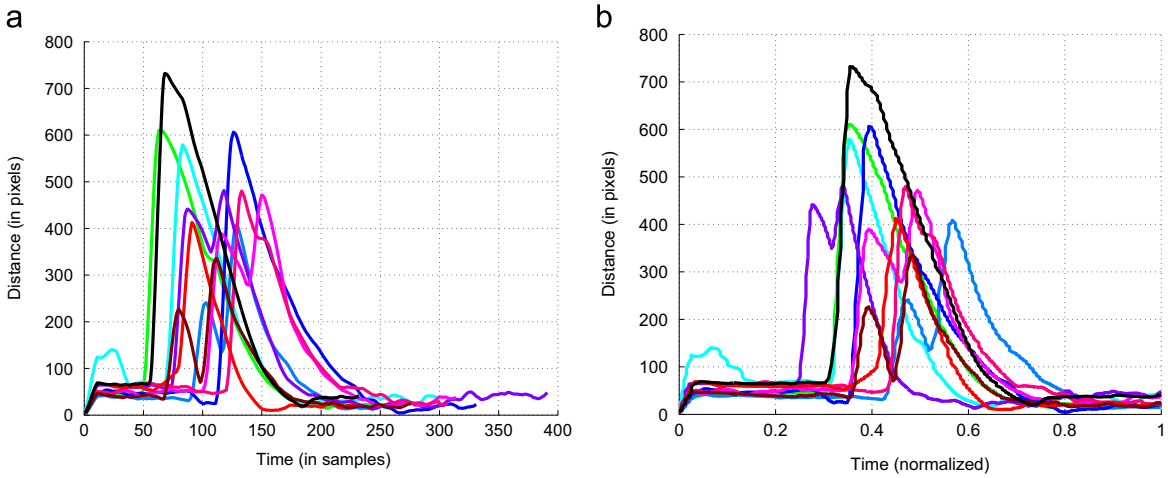

Fig. 8. Sketch–gaze distance curves corresponding to 10 repeated task instances of a user each drawn in a distinguishing color. (a) Original sketch–gaze distance curves. (b) Normalized sketch–gaze distance curves. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

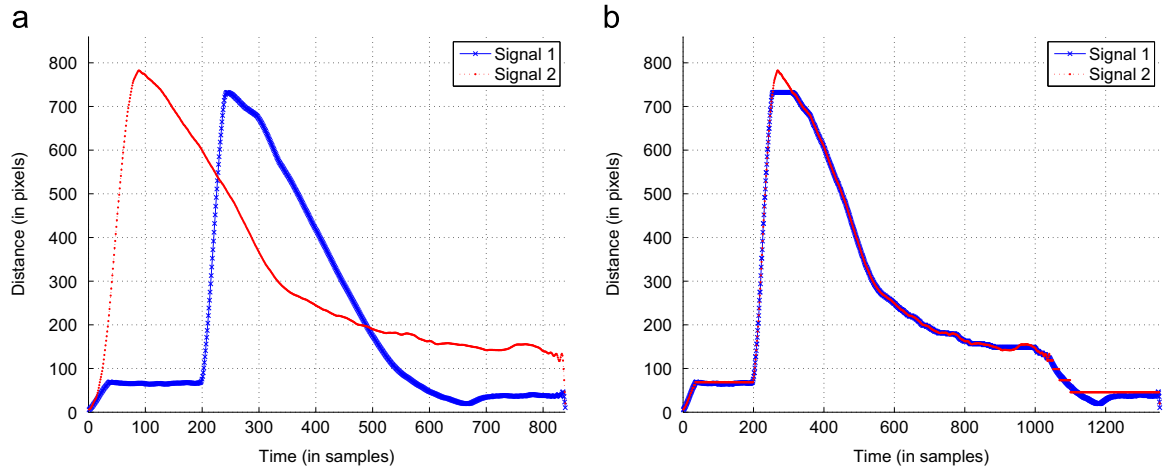

Fig. 9. Sketch–gaze distance curves corresponding to two task instances of a user. We use dynamic time warping for computing an optimal alignment between two given curves by warping each curve with respect to the other one. (a) Original sketch–gaze distance curves. (b) Warped sketch–gaze distance curves.

each virtual interaction task using sketch–gaze distance curves of all task instances. These task-specific sketch–gaze distance curves will be referred to as *characteristic curves*.

4.1.2. Characteristic curve extraction

Fig. 8a illustrates the sketch–gaze distance curves corresponding to 10 repeated task instances of a user for the *drag* task in the large scale. These curves have been filtered by a symmetric Gaussian low-pass filter of size 11×1 and $\sigma = 5$. It is evident that the user naturally spent different amounts of time to complete each task instance. In order to overcome the discrepancy in task completion times, sketch–gaze distance curves are normalized with respect to a standard time range as depicted in Fig. 8b. However, even after the normalization procedure, the sketch–gaze distance curves are still not sufficiently aligned. This indicates that although users accomplish similar sub-tasks to complete each task, the completion time and speed of these sub-tasks vary across task instances, and even within a user.

To align sketch–gaze distance curves that are similar in shape but evolve at different rates, we use *dynamic time warping* (Sakoe and Chiba, 1978). Dynamic time warping is a sequence alignment method often used in the time series classification domain to measure the similarity between two sequences independent of non-linear variations in the time dimension. We use it both for computing the similarity of two given curves and for finding an optimal alignment between them. Fig. 9 demonstrates the alignment of two curves using dynamic time warping.[3] Alternative sequence alignment methods include, but are not limited to, functional data analysis (Ramsay, 2006), curve alignment by moments (James, 2007) and curve synchronization based on structural characteristics (Kneip and Gasser, 1992).

We build scale- and task-specific characteristic curves as follows:

1. For each task instance, we obtain the instantaneous sketch–gaze distance curve D_i.
2. We smooth out all D_i by a rotationally symmetric Gaussian low-pass filter of size 11×1 and $\sigma = 5$.
3. We form a similarity matrix S based on the similarity values corresponding to all possible pair combinations of sketch–gaze distance curves. Similarity values are computed using the dynamic time warping algorithm.
4. We create a hierarchical cluster tree from the similarity matrix. Clusters are computed using the unweighted pair group method with arithmetic mean (UPGMA) based on the Euclidean distance metric.

[3] We use an open-source dynamic time warping library for MATLAB (Felty, 2004).

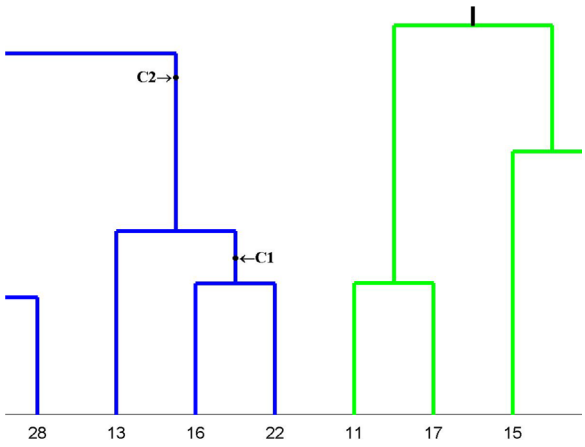

Fig. 10. Weighted hierarchical time warping algorithm. According to this algorithm, the curve labeled *C1* is created by warping the curves with indices 16 and 22 whereas the curve labeled *C2* is created by warping the curve with index 13 and the previously created *C1* curve. Here, $C1 = \frac{1}{2} \times dtw(16, 22) + \frac{1}{2} \times dtw(22, 16)$ whereas $C2 = \frac{1}{3} \times dtw(13, C1) + \frac{2}{3} \times dtw(C1, 13)$. Note that *dtw(source, target)* is the dynamic time warping function that warps the source curve with respect to the target curve and returns the warped source curve. The weights determine how much the warped source curve contributes to the final warping result.

5. On each cluster, an algorithm we call *weighted hierarchical time warping* is applied. We developed this algorithm for computing the characteristic curve that best represents any given cluster of curves. The weight of an input curve depends on the number of leaves below the node corresponding to the input curve in the hierarchical cluster dendrogram. This way, all members of a cluster contribute equally to the resulting characteristic curve. The details of this algorithm are described in Fig. 10.
6. We take the final weighted hierarchical warping result of the cluster with the maximum number of elements as the characteristic curve of the respective task and scale. If there are multiple clusters containing at least 10 task instances, then each of these clusters contributes a characteristic curve to the set of characteristic curves for the respective task and scale.

Algorithm 1. Algorithm for building the instantaneous sketch–gaze distance curves.

Input: Gaze data G_i and sketch data P_i for all task instances of the input task and scale
Output: Instantaneous sketch–gaze distance curves D'_i
1: **for all** Task instances i **do**
2: $D_i \leftarrow |G_i - P_i|$
3: $D'_i \leftarrow smooth(D_i)$
4: **end for**

Algorithm 2. Algorithm for forming the similarity matrix.

Input: D'_i for all task instances of the input task and scale
Output: Similarity matrix S
1: **for all** Task instance pairs i, j **do**
2: $S_{ij} \leftarrow dtw\_distance(D'_i, D'_j)$ ◃ dtw_distance method computes the similarity of two given curves.
3: **end for**

Algorithm 3. Algorithm for extracting the characteristic curve(s).

Input: Similarity matrix S of the input task and scale
Output: An array of characteristic curve(s)

1: $clusterArray \leftarrow linkage(S)$
2: **for all** clusters C_i in *clusterArray* with at least 10 task instances **do**
3: initialize $w_j \leftarrow 1$ for all members j of cluster C_i
4: **while** C_i contains multiple curves
5: find the most similar curve pair (*first*, *second*) in the cluster
6: warp the first curve with respect to the second curve to get *firstWarped*
7: warp the second curve with respect to the first curve to get *secondWarped*
8: $firstWeight \leftarrow \frac{w_{first}}{w_{first} + weight_{second}}$ and $secondWeight \leftarrow \frac{w_{second}}{w_{first} + w_{second}}$
9: take a weighted average of the warped curves to get *newCurve*
10: $w_{newCurve} \leftarrow w_{first} + w_{second}$
11: replace the warped curves in the cluster with the newly computed curve
12: **end while**
13: add the final *newCurve* to the array of characteristic curves
14: **end for**

Using this algorithm, we obtain a characteristic curve for each task and scale as depicted in Fig. 11 for the large scale. Given a sketch–gaze distance curve, we construct its feature vector by measuring its similarity to each of these characteristic curves. This vector corresponds to the first feature of our novel gaze-based feature representation. Again, similarity values are calculated using dynamic time warping. Although not shown in this figure, some tasks may have multiple characteristic curves. This happens if there exist multiple strategies that users follow to complete a specific task. Therefore, the length of this feature vector is not constant and depends on the total number of characteristic curves.

A qualitative investigation of the characteristic curves brings up interesting observations on stylus-gaze coordination behavior. In line with our initial argument, this behavior is observed to be task-dependent. Furthermore, the characteristic curves have easily interpretable shapes. For instance, in the *scroll* task, the hand keeps pulling the chain while the eyes are busy attending to the newly appearing information on the display. Therefore, one would expect the distance between the hand and the eyes to increase constantly; our findings agree with this expectation (Fig. 11).

4.2. Feature 2: within-cluster variance of gaze positions

As we demonstrate in the next subsection, eye gaze positions along the path of different virtual interaction tasks exhibit different clustering behaviors. Hence, a measure of how the gaze points are clustered and spread out along the trajectory of the task carries discriminative information for task identification. This is what we attempt to capture with the within-cluster variance feature.

4.2.1. The rationale

Humans employ two different modes of voluntary gaze-shifting mechanism to orient the visual axis. These modes are referred to as saccadic and smooth pursuit eye movements. It is widely accepted that "saccades are primarily directed toward stationary targets whereas smooth pursuit is elicited to track moving targets" (de Xivry and Lefèvre, 2007). Typical virtual interaction tasks contain both stationary and moving targets. A user's attention can be dominantly directed towards targets of either type depending on the intended task.

Our experiments show that in a typical *drag* task, saccades are more common and the user's attention is drawn from one stationary target which is the initial position of the object

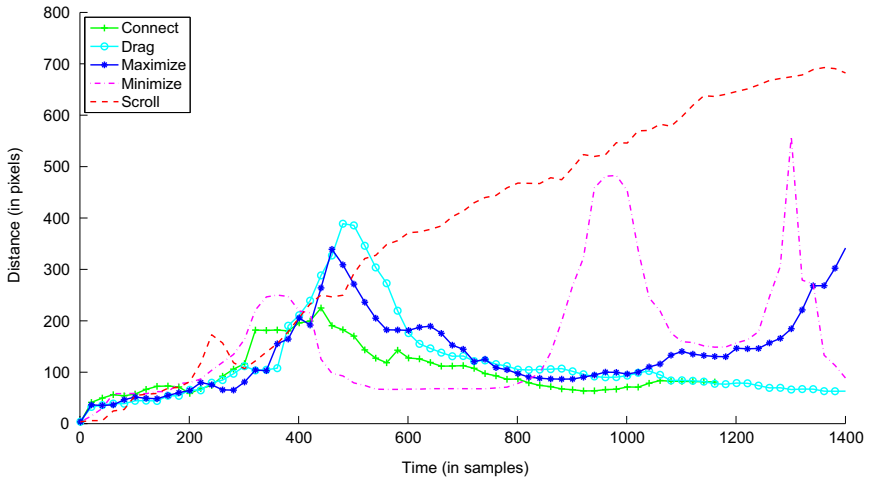

Fig. 11. Characteristic curves obtained from sketch–gaze distance curves of each task in large scale.

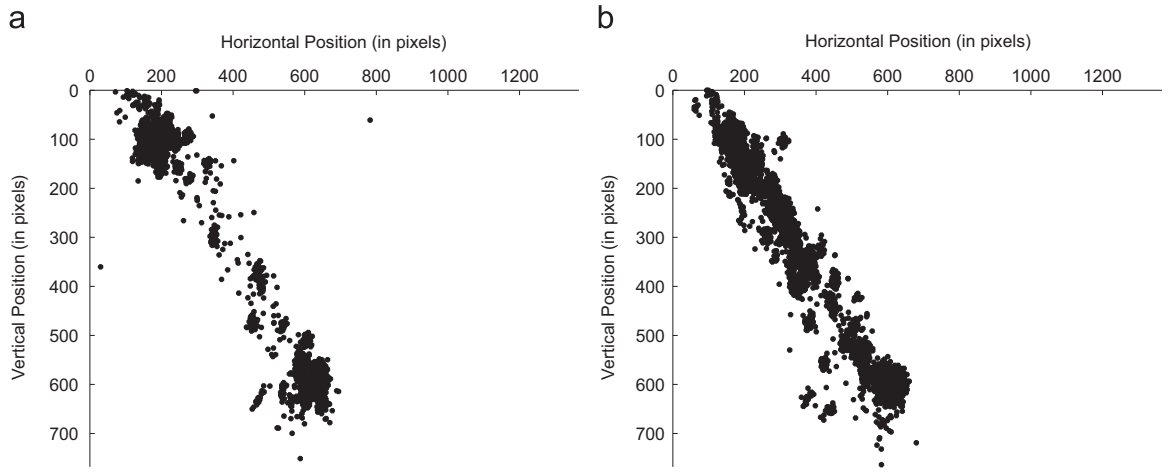

Fig. 12. Gaze data corresponding to 10 repeated task instances of a user. (a) Gaze data for the *drag* task. Saccadic eye movements result in gaze point clusters with low within-cluster variance. (b) Gaze data for the *free-form drawing* task. Smooth pursuit eye movements result in gaze point clusters with high within-cluster variance.

currently being dragged to the other stationary target which is the intended position of the object (Fig. 12a). Conversely during *free-form drawing* (Fig. 12b), smooth pursuit is more common and the user's attention is drawn to the moving target (the newly appearing ink). In saccades, gaze points accumulate around the stationary targets whereas in smooth pursuit, gaze points scatter along the pursuit path. The second feature of our novel gaze-based feature representation is based on these observations, and hence attempts to quantify how the eye gaze data is structured in terms of saccades and fixations.

4.2.2. *Quantifying the distribution of saccades and fixations*

We quantify the distribution of saccades and fixations by measuring the mean within-cluster variance of clustered gaze points for each task instance. Clustering is done via MATLAB's *k-means clustering* algorithm and repeated three times for different k values as $k = 1, 2, 3$. Thus, the length of this feature vector is constant and equal to 3. We do not use higher orders of k since gaze packets aimed at the source and target objects respectively form the first and second clusters while the remaining gaze packets form the third cluster.

5. Intention prediction and evaluation

In this section, we evaluate the effectiveness of the features introduced above in predicting virtual interaction tasks. During evaluation, we focus on several aspects, including the prediction accuracy and scale-invariance. In addition, we run feature selection tests to evaluate the relevance and redundancy of the features introduced above. We also compare the prediction power of our novel gaze-based feature representation to that of commonly utilized and well-established sketch-based feature representations in the literature.

As mentioned earlier in Section 3, we record participants' eye gaze as well as the pen trajectory during the execution of each task instance. A subset of sketch data from our database is shown in Fig. 13. As seen in Fig. 13, even though the individual pen trajectories for our tasks do not appear to be as stylized as those in Fig. 5, it is still conceivable that pen trajectories alone may suffice for accurate task prediction. To this end, we experimented with a number of image-based approaches to extract features from the collected sketch data. These feature representations, IDM Features (Ouyang and Davis, 2009) and Zernike Moments (Khotanzad and Hong, 1990), are shown to work well for hand-drawn sketch data by Tümen et al. (2010). The authors further demonstrate that to achieve good recognition accuracies with these feature representations, good feature extraction

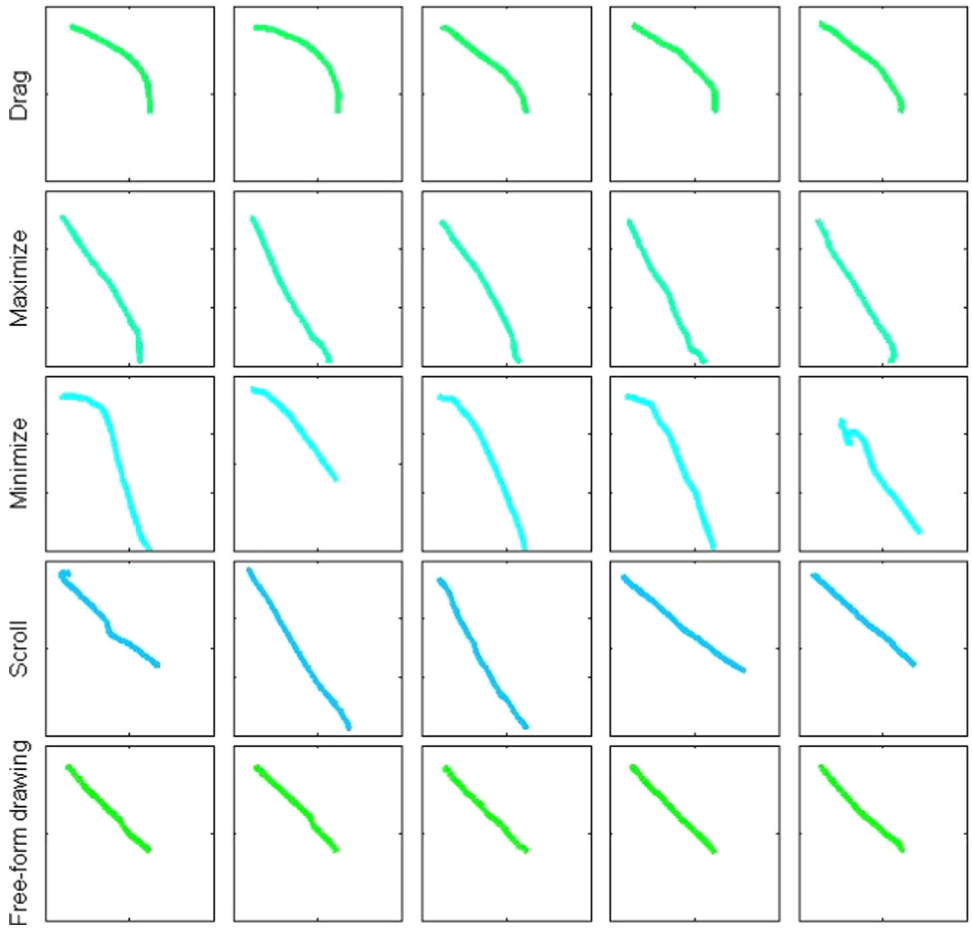

Fig. 13. Sketch data corresponding to a user's 5 repeated task instances for 5 tasks. Pen trajectories for our tasks serve as an example for non-stylized pen input that do not have easily distinguishable characteristic visual appearance.

Table 3
Parameter settings for the sketch-based feature representations.

Feature representation	Parameter settings
IDM features	$k=25$, $\sigma=3$ and $r=150$
Zernike moments	$o=12$

parameters must be selected. IDM Features have three free feature extraction parameters as k (kernel size), σ(smoothing factor) and r (resampling parameter). Zernike Moments have one free parameter, which is the order of the Zernike moment o. We set the parameters of the evaluated sketch-based feature representations in accordance with the optimum values reported in Tümen et al. (2010). For reproducibility, our parameter settings are listed in Table 3.

All accuracy tests were done using the LIBSVM (Chang and Lin, 2011) implementation of Support Vector Machines. The accuracies are measured in line with the standard three-step machine learning pipeline, where we first extract feature vectors from a set of data samples, then train classifier models using these feature vectors, and finally measure accuracies using unseen data.

1. We partition the input data into 5 disjoint folds, chosen randomly but with roughly equal size. Out of these 5 folds, 4 are reserved for training and validating the model whereas the remaining fold is reserved for testing the model.
2. We extract feature vectors from the training data, and normalize them by standardization.
3. We train a model using the Gaussian radial basis function (RBF) kernel. We estimate the hyper-parameters of our model using grid search with 5-fold cross-validation.
4. We evaluate the resulting prediction model on the testing data to obtain accuracy.
5. Steps 2–4 are repeated for each random split in a round-robin fashion such that each of the 5 folds is used exactly once for testing.
6. The mean accuracy for the random splits is reported.

5.1. Accuracy tests

Our accuracy tests fall under two categories: The first set of tests evaluates gaze-based and sketch-based feature representations individually, and second set evaluates their combinations. The accuracy tests are carried out and reported for the *large*, *medium* and *small* scales, as well as for the *all scales* case, which corresponds to the entire database.

Collectively, the results of the individual tests suggest that feature representation has an effect on prediction accuracy. Specifically, our results suggest that the gaze-based feature representation is significantly better in capturing the richness and complexity of our user input when compared to various sketch-based feature representations that have been shown to work well for hand-drawn sketch data. On the other hand, results of the combined tests indicate that combining gaze-based and sketch-based feature representations may yield higher accuracy

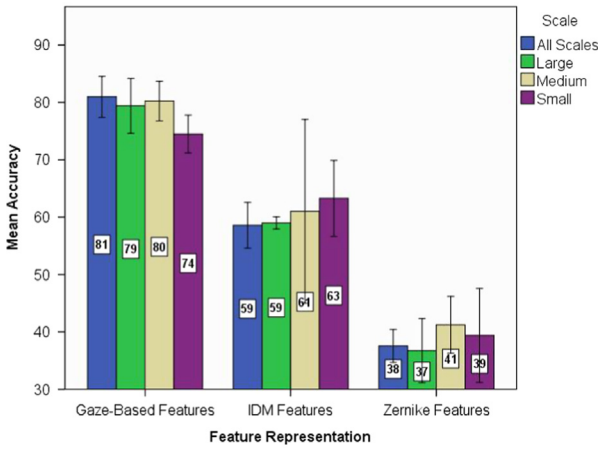

Fig. 14. Mean accuracy scores for each feature representation and scale. Error bars indicate 95% confidence interval.

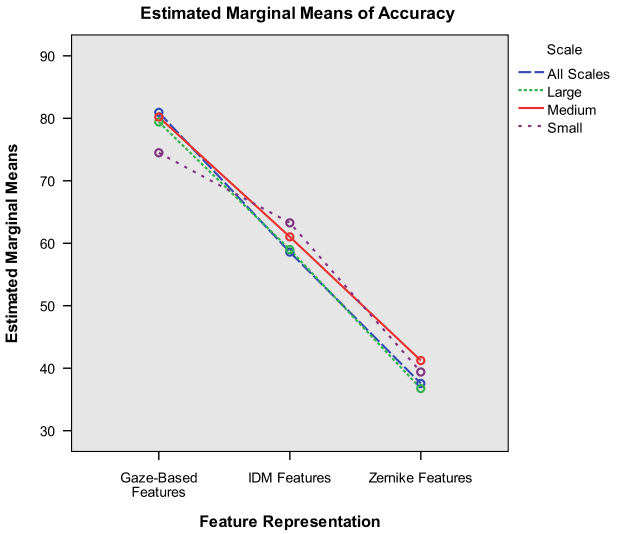

Fig. 15. Two-way ANOVA results that examine the interaction of feature representation and scale factors on prediction accuracy.

scores depending on the choice of sketch-based feature representation and the combination technique.

5.1.1. Accuracy tests for evaluating the feature representations individually

Fig. 14 shows the mean accuracies for individual feature representations. We conducted a two-way ANOVA to examine the effect of feature representation and scale on prediction accuracy. ANOVA revealed a main effect of feature representation on prediction accuracy across the *Gaze-Based Features* (78.76 ± 3.84), *IDM Features* (60.46 ± 6.86) and *Zernike Moments* (38.73 ± 4.59) conditions at the $p < 0.05$ level, [$F(2, 48) = 292.924$, $p < 0.001$]. Post-hoc comparisons using the Tukey HSD test indicated that the mean score for the *Gaze-Based Features* condition was significantly higher than the *IDM Features* condition ($p < 0.001$) and the *Zernike Moments* condition ($p < 0.001$). In addition, the mean score for the *IDM Features* condition was found to be significantly higher than the *Zernike Moments* condition ($p < 0.001$).

There was no main effect of scale on prediction accuracy across the *large* (58.37 ± 18.32), *medium* (60.82 ± 18.04), *small* (59.04 ± 15.88) and *all scales* (59.03 ± 18.53) conditions at the $p < 0.05$ level, [$F(3, 48) = 0.602, p = 0.617$]. This indicates that there is not enough evidence to show that our prediction system has a significantly higher/lower accuracy score for any particular scale irrespective of feature representation. Furthermore, there was no significant interaction between feature representation and scale, [$F(6, 48) = 1.268, p = 0.290$]. In other words, we can infer that there is not enough evidence to show that a particular feature representation performs significantly better or worse under scale variations. Fig. 15 provides a graphical illustration of the interactions.

5.1.2. Accuracy tests for evaluating combinations of feature representations

The individual accuracy tests focus on the performance of individual feature representations. A natural follow-up to the previous experiments is to explore whether gaze-based and sketch-based feature representations can be combined to increase prediction accuracy. There are two common techniques for information fusion, namely classifier-level fusion and feature-level fusion. Mean accuracy values computed for each scale with all possible classifier-level fusion and feature-level fusion combinations of the gaze-based and sketch-based feature representations are shown in Fig. 16.

Classifier-Level Fusion: For classifier-level fusion, we train two probabilistic SVM models – one with the gaze-based features and another with either of the sketch-based features (IDM Features or

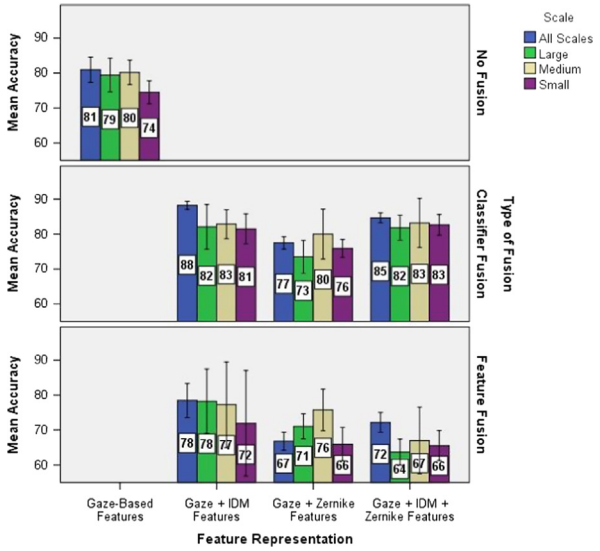

Fig. 16. Accuracy tests with the classifier-level fusion and feature-level fusion techniques. Error bars indicate 95% confidence interval.

Zernike Moments). The output of each probabilistic SVM model is a vector of size 5, each element of the vector representing the probability estimate of the input sample being a member of the five respective virtual task classes. We then use the outputs of these two probabilistic SVM models to train a third multi-class SVM model. The feature vector in this case is a vector of size 10 consisting of probability estimate values from the gaze-based and sketch-based probabilistic SVM models, respectively.

Feature-Level Fusion: For feature-level fusion, feature vectors corresponding to multiple feature representations are concatenated to construct a high-dimensional feature vector. Then, a regular SVM model is trained with this feature vector.

Statistical analysis of the accuracy tests with the combination of gaze-based and sketch-based feature representations imply the following results (For brevity, we take $p = 0.05$ unless otherwise noted.):

- Classifier-level fusion of Gaze-Based Features and IDM Features (83.66 ± 4.28) gives the overall highest mean accuracy value (see Fig. 17a).

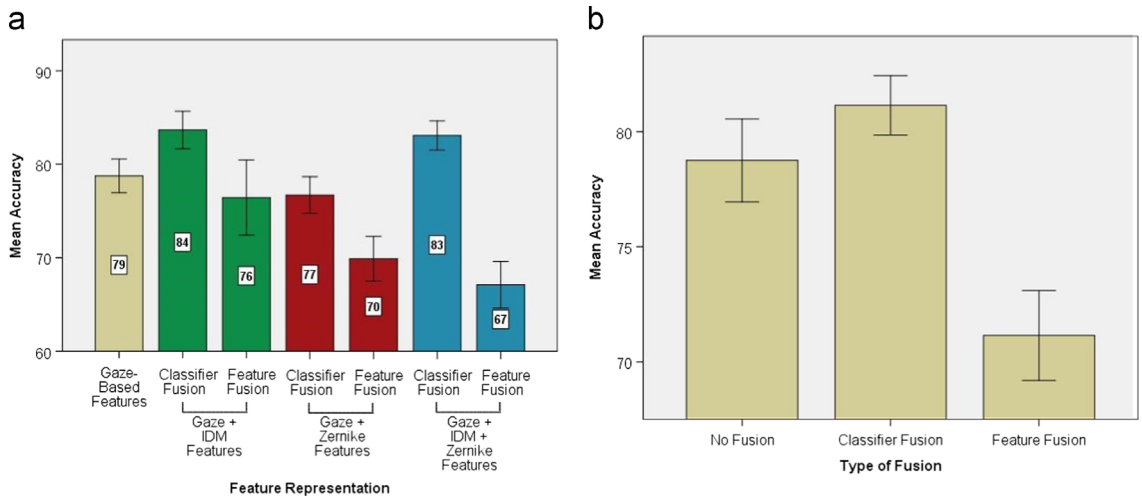

Fig. 17. Summary of results for the combined accuracy tests. Error bars indicate 95% confidence interval. (a) Mean accuracy values for various combinations of feature representations and information fusion techniques. (b) Mean accuracy values for the no fusion, classifier-level fusion and feature-level fusion cases. No fusion case corresponds to using Gaze-Based Features alone.

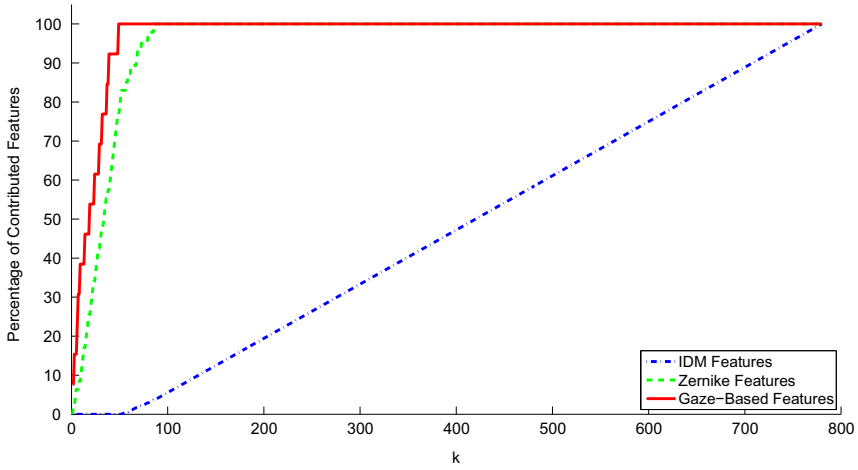

Fig. 18. Percentage of contributed features by each feature representation to the best performing set of features selected by the mRMR framework.

- IDM Features give higher mean accuracy values than Zernike Moments in both classifier- (83.66 ± 4.28 vs. 76.72 ± 4.16) and feature-level fusion (76.44 ± 8.59 vs. 69.89 ± 5.16) cases (see Fig. 17a).
- Classifier-level fusion (81.15 ± 5.01) yields higher mean accuracy values compared to feature-level fusion (71.14 ± 7.55) (see Fig. 17b).
- Gaze-Based Features alone (i.e. no fusion) (78.76 ± 3.84) give higher mean accuracy values compared to feature-level fusion technique (71.14 ± 7.55) (see Fig. 17b).

5.2. *Feature selection tests for evaluating the relevance and redundancy of the feature representations*

Our accuracy tests show that combining the Gaze-Based Features and IDM Features by classifier-level fusion gives the overall highest mean accuracy value. However, in practice, it might not be feasible to extract hundreds of features in real-time. In that case, we can use feature selection to obtain a faster and cost-effective predictor by ranking the features based on a mutual information criterion and selecting a feasibly smaller subset of the highly ranked features. This subset is composed of the maximally relevant and minimally redundant (i.e. the best performing) features selected among all feature representations in consideration.

Feature selection tests were conducted within the Maximum Relevance & Minimum Redundancy (mRMR) feature selection framework (Peng et al., 2005). This framework allows us to select the k maximally relevant and minimally redundant features from a total set of K features where $k \leq K$. In our case, respective lengths of feature vectors generated by Gaze-Based Features, IDM Features and Zernike Features are $f_1 = 13, f_2 = 720$ and $f_3 = 47$. Therefore the total number of features is $K = f_1 + f_2 + f_3 = 780$. Fig. 18 shows the percentage of features contributed by each feature representation to the best performing set of features. As seen here, Gaze-Based Features surpass (or equal) both sketch-based feature representations in terms of the percentage of contributed features for all values of k. All features generated by the gaze-based feature representation make their way into the best performing set of features by $k = 49$. At this point, only as little as 6.28% of the total number of features are used. Therefore, in cases where speed and cost are of concern, Gaze-Based Features offer a better alternative to IDM Features and Zernike Features.

5.3. *Scale-invariance tests*

Practical usage of our prediction system may involve a range of display devices and user interfaces with varying sizes and

constraints. Robustness of a feature representation to variations in scale is important if we want our prediction system to work equally accurately despite these variations. Fig. 14 shows the mean accuracies for individual feature representations in different scales. We previously referred to this figure in Section 5.1 for our accuracy tests, but we did not focus on the scale-invariance of our task prediction system. As we substantiate in detail in the next two subsections, our task prediction system is scale-invariant in all Gaze-Based Features, IDM Features and Zernike Features cases. The only exception is for pen and eye movements that are entirely in the foveal area. In that case, prediction accuracy deteriorates by a small, yet statistically significant amount for the gaze-based feature representation. This is expected due to limitations of our eye tracker in smaller scales in terms of tracking accuracy.

5.3.1. Scale-invariance tests with the gaze-based feature representation

In order to compare the effect of scale on prediction accuracy across the *large* (79.40 ± 3.85), *medium* (80.20 ± 2.79), *small* (74.47 ± 2.66) and *all scales* (80.95 ± 2.89) conditions, we conducted a one-way between subjects ANOVA with the gaze-based feature representation. There was a significant effect of scale on prediction accuracy at the $p < 0.05$ level for the four conditions [$F(3,16) = 4.497, p = 0.018$]. Post-hoc comparisons using the Tukey HSD test indicate that the mean score for the *small* condition is significantly lower than the *all scales* condition ($p = 0.020$) and the *medium* condition ($p = 0.043$). However, there were no differences between the *all scales*, *large* and *medium* conditions. More specifically, $p = 0.855$ for *all scales* and *large* conditions, $p = 0.980$ for *all scales* and *medium* conditions, and finally $p = 0.976$ for *large* and *medium* conditions.

Collectively, these results suggest that length of the task trajectory has an effect on prediction accuracy. Specifically, our results indicate that when the range of pen and eye movements in a task approaches the range of foveal human vision, prediction accuracy deteriorates slightly. This is expected, because tracking error is relatively worse for smaller scales. Wider ranges of pen and eye movements do not appear to increase or decrease prediction accuracy significantly. On the other hand, the most realistic test condition corresponds to the *all scales* case since data collected during natural interaction with a user interface will typically consist of tasks across a variety scales. Prediction accuracy in fact peaks at the *all scales* case.

5.3.2. Scale-invariance tests with the sketch-based feature representations

In order to compare the effect of scale on prediction accuracy across the *large* ($58.98 \pm 0.85/36.74 \pm 4.51$), *medium* ($61.02 \pm 12.90/41.23 \pm 3.99$), *small* ($63.27 \pm 5.35/39.39 \pm 6.60$) and *all scales* ($58.57 \pm 3.20/37.55 \pm 2.31$) conditions, we conducted a one-way between subjects ANOVA with IDM Features/Zernike Moments. For both feature representations, there was no significant effect of scale on prediction accuracy at the $p < 0.05$ level for the four conditions, more specifically [$F(3,16) = 0.451, p = 0.720$] for IDM Features and [$F(3,16) = 0.942, p = 0.444$] for Zernike Moments.

6. Future work and concluding remarks

We have proposed a gaze-based virtual task prediction system to alleviate dependence on explicit mode switching in pen-based systems. Our system infers intended user actions by monitoring and analyzing eye gaze movements that users naturally exhibit during pen-based user interaction. More specifically, our system successfully discriminates between frequently employed pen-based virtual manipulation commands: *drag*, *maximize*, *minimize* and *scroll*. In addition, our system differentiates between the intention to sketch and the intention to issue a command. We believe that predicting the mode of interaction will eventually allow us to build systems that save users the trouble of mode switching during basic interaction tasks.

Our first contribution is a carefully compiled multimodal dataset that consists of eye gaze and pen input collected from participants completing various virtual interaction tasks. Our second contribution is a novel gaze-based feature representation, which is rooted in our understanding of human perception and gaze behavior. Our feature representation is neither subject- nor interface-specific, and performs better than common, well-established sketch recognition feature representations in the literature. Our third contribution is a novel gaze-based task prediction system based on this feature representation that can generalize to variations in task type and scale. The prediction results that we report are substantially better than existing work in the literature that attempt multi-class intention prediction as we do. Furthermore, we do not require defining application and interface specific areas of interests.

In the light of promising findings reported in this paper, we envision a number of immediate follow-ups to our work, as well as long term research directions to explore. An immediate extension might involve conducting experiments to see if our prediction framework is equally suitable to other pointer-based user interfaces rather than being limited to pen-based user interfaces only. Another possible direction might involve conducting experiments to see if our prediction system can successfully recognize other virtual tasks.

More substantial extensions might explore if variants of a particular virtual task can be discriminated. For example, it is conceivable that a minimization task where the target size is set in reference to another virtual object may result in different stylus-gaze behavior compared to the case without a reference object. This may involve building a finer taxonomy of virtual tasks (e.g. drag with/without a target, minimize with/without a reference, etc.), and extending the feature representation to handle these finer distinctions.[4]

Other long term follow-up work might explore the feasibility of using our prediction system to build a proactive user interface. We envision a proactive system capable of actively monitoring user's gaze and pen input to detect the intention to switch modes in an online setting, and act accordingly. We believe that we can successfully combine the idea of online prediction with uncertainty visualization, gaze-contingent rendering (Duchowski et al., 2004) and transparent layered user interfaces (Harrison et al., 1995) to build online prediction systems that avoid the Midas touch problem. Further experiments would be required to evaluate the usability aspects of this setup, and compare it to the state of the art mode switching mechanisms in the literature. Another extension might explore the suitability of our feature representation scheme to aid the recognition and segmentation of sketches (e.g. URL diagrams, circuit diagrams). It is conceivable that conceptually different subtasks of sketching – such as drawing objects, connectors, arrows or producing handwritten annotations – may each have distinct gaze-stylus interactions, which might be captured by extending feature representations introduced in this paper.

Acknowledgements

The authors gratefully acknowledge the support and funding of TÜBİTAK (The Scientific and Technological Research Council of Turkey) under grant number 110E175 and TÜBA (Turkish Academy of Sciences).

[4] In fact, we believe existing categorization of virtual tasks is rather coarse, and there is a pressing need to construct a fine grained taxonomy that highlights the differences between various flavors of these tasks.

Appendix A. Related gaze-based task prediction approaches

Paper Title	Primary Author	Year	Analyze/Predict	Daily/Virtual	Tasks
In what ways do eye movements contribute to everyday activities?	Micheal F. Land	2001	Analyze	Daily	Tea making Sandwich making
A robust algorithm for reading detection	Christopher S. Campbell	2001	Predict	Virtual	Reading Skimming Scanning
Understanding human behaviors based on eye-head-hand coordination	Chen Yu	2002	Predict	Daily	Unscrewing a jar Stapling a letter Pouring water
Learning to recognize human action sequences	Chen Yu	2002	Predict	Daily	Stapling a letter
Using eye gaze patterns to identify user tasks	Shamsi Tamara Iqbal	2004	Analyze	Virtual	Reading comprehension Mathematical reasoning Search Object manipulation
Eye movements in natural behavior	Mary Hayhoe	2005	Analyze	Daily	Everyday visually guided behaviors
Eye and pen: a new device for studying reading during writing	Denis Alamargot	2006	Analyze	Virtual	Reading Writing
Human gaze behavior during action execution and observation	Benno Gesierich	2008	Analyze	Virtual	Action execution Action observation
Recognizing behavior in hand-eye coordination patterns	Weilie Yi	2009	Analyze	Daily	10 subtasks of a sandwich making task
Multimodal integration of natural gaze behavior for intention recognition during object manipulation	Thomas Bader	2009	Predict	Virtual	Object manipulation
Eye movement analysis for activity recognition	Andreas Bulling	2009	Predict	Daily	Copy Read Write Video Browse Null
What's in the eyes for context-awareness?	Andreas Bulling	2011	Predict	Daily	- Reading or not reading - Copy, read, write, video, browse, null - Visual memory (Familiar/unfamiliar images)
Activity recognition using eye-gaze movements and traditional interactions	François Courtemanche	2011	Predict	Virtual	Subtasks of Google Analytics and eLearning tasks
Learning to recognize daily actions using gaze	Alireza Fathi	2012	Predict	Daily	Meal preparation
What do you want to do next: a novel approach for intent prediction in gaze-based interaction	Roman Bednarik	2012	Predict	Virtual	Issue a command or not
Coupling eye-motion and ego-motion features for first-person activity recognition	Keisuke Ogaki	2012	Predict	Daily	Copy Read Write Video Browse Null
User-adaptive information visualization – Using eye gaze data to infer visualization tasks and user cognitive abilities	Ben Steichen	2013	Predict	Virtual	Retrieve value Filter Compute derived value Find extremum Sort

References

Alamargot, D., Chesnet, D., Dansac, C., Ros, C., 2006. Eye and pen: a new device for studying reading during writing. Behav. Res. Methods 38 (2), 287–299.

Bader, T., Vogelgesang, M., Klaus, E., 2009. Multimodal integration of natural gaze behavior for intention recognition during object manipulation. In: Proceedings of the Eleventh International Conference on Multimodal Interfaces, ACM, New York, NY, USA, pp. 199–206.

Ballard, D.H., Hayhoe, M.M., Li, F., Whitehead, S.D., Frisby, J.P., Taylor, J.G., Fisher, R. B., 1992. Hand-eye coordination during sequential tasks [and discussion]. Philos. Trans. Biol. Sci. 337 (1281), 331–339.

Bednarik, R., Vrzakova, H., Hradis, M., 2012. What do you want to do next: a novel approach for intent prediction in gaze-based interaction. In: Proceedings of the Symposium on Eye Tracking Research and Applications, ACM, New York, NY, USA, pp. 83–90.

Bulling, A., Ward, J.A., Gellersen, H., Tröster, G., 2009. Eye movement analysis for activity recognition. In: Proceedings of the Eleventh International Conference on Ubiquitous Computing, ACM, New York, NY, USA, pp. 41–50.

Bulling, A., Roggen, D., Tröster, G., 2011. What's in the eyes for context-awareness? Pervasive Computing, IEEE 10 (2), 48–57.

Campbell, C.S., Maglio, P.P., 2001. A robust algorithm for reading detection. In: Proceedings of the 2001 Workshop on Perceptive User Interfaces, ACM, New York, NY, USA, pp. 1–7.

Chang, C.-C., Lin, C.-J., 2011. Libsvm: a library for support vector machines. ACM Trans. Intell. Syst. Technol. 2 (3), 1–27.

Courtemanche, F., Aïmeur, E., Dufresne, A., Najjar, M., Mpondo, F., 2011. Activity recognition using eye-gaze movements and traditional interactions. Interact. Comput. 23 (3), 202–213.

de Xivry, J.-J.O., Lefèvre, P., 2007. Saccades and pursuit: two outcomes of a single sensorimotor process. J. Physiol. 584 (1), 11–23.

Duchowski, A.T., Cournia, N., Murphy, H.A., 2004. Gaze-contingent displays: a review. Behav. Social Netw. 7 (6), 621–634.

Fathi, A., Li, Y., Rehg, J.M., 2012. Learning to recognize daily actions using gaze. In: Proceedings of the Twelfth European Conference on Computer Vision – Volume Part I, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 314–327.

Felty, T., 2004. Dynamic time warping (Dec. 2004). ⟨http://www.mathworks.com/matlabcentral/fileexchange/6516-dynamic-time-warping/⟩.

Forlines, C., Balakrishnan, R., 2008. Evaluating tactile feedback and direct vs. indirect stylus input in pointing and crossing selection tasks. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, New York, NY, USA, pp. 1563–1572.

Gesierich, B., Bruzzo, A., Ottoboni, G., Finos, L., 2008. Human gaze behaviour during action execution and observation. Acta Psychol. 128 (2), 324–330.

Harrison, B.L., Ishii, H., Vicente, K.J., Buxton, W.A.S., 1995. Transparent layered user interfaces: an evaluation of a display design to enhance focused and divided attention. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 317–324.

Hayhoe, M., Ballard, D., 2005. Eye movements in natural behavior. Trends Cogn. Sci. 9 (4), 188–194.

Iqbal, S.T., Bailey, B.P., 2004. Using eye gaze patterns to identify user tasks. In: The Grace Hopper Celebration of Women in Computing, pp. 5–10.

James, G.M., 2007. Curve alignment by moments. Ann. Appl. Stat. 1 (2), 480–501.

Johansson, R.S., Westling, G., Bäckström, A., Flanagan, J.R., 2001. Eye-hand coordination in object manipulation. J. Neurosci. 21 (17), 6917–6932.

Khotanzad, A., Hong, Y.H., 1990. Invariant image recognition by zernike moments. IEEE Trans. Pattern Anal. Mach. Intell. 12 (5), 489–497.

Kneip, A., Gasser, T., 1992. Statistical tools to analyze data representing a sample of curves. Ann. Stat. 20 (3), 1266–1305.

Kumar, M., Paepcke, A., Winograd, T., 2007. Eyepoint: practical pointing and selection using gaze and keyboard, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, New York, NY, USA, pp. 421–430.

Land, M.F., Hayhoe, M., 2001. In what ways do eye movements contribute to everyday activities? Vision Res. 41 (25–26), 3559–3565.

Li, Y., Hinckley, K., Guan, Z., Landay, J.A., 2005. Experimental analysis of mode switching techniques in pen-based user interfaces. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, New York, NY, USA, pp. 461–470.

Negulescu, M., Ruiz, J., Lank, E., 2010. Exploring usability and learnability of mode inferencing in pen/tablet interfaces. In: Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium. Eurographics Association, Aire-la-Ville, Switzerland, pp. 87–94.

Nielsen, J., 1993. Noncommand user interfaces. Commun. ACM 36 (4), 83–99.

Ogaki, K., Kitani, K.M., Sugano, Y., Sato, Y., 2012. Coupling eye-motion and ego-motion features for first-person activity recognition. In: Computer Vision and Pattern Recognition Workshops, IEEE, pp. 1–7.

Ouyang, T.Y., Davis, R., 2009. A visual approach to sketched symbol recognition. In: Proceedings of the Twenty-first International Joint Conference on Artifical Intelligence, pp. 1463–1468.

Peng, H., Long, F., Ding, C., 2005. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans. Pattern Anal. Mach. Intell. 27, 1226–1238.

Plimmer, B., 2008. Experiences with digital pen, keyboard and mouse usability. J. Multimodal User Interfaces 2 (1), 13–23.

Ramsay, J.O., 2005. Functional Data Analysis. Springer, New York, NY, USA.

Rayner, K., 2009. Eye movements and attention in reading, and visual search. Q. J. Exp. Psychol. 62 (8), 1457–1506.

Rubine, D., 1991. Specifying gestures by example. SIGGRAPH Comput. Graph. 25 (4), 329–337.

Sakoe, H., Chiba, S., 1978. Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. Acoust. Speech Signal Process. 26 (1), 43–49.

Steichen, B., Carenini, G., Conati, C., 2013. User-adaptive information visualization: using eye gaze data to infer visualization tasks and user cognitive abilities, in: Proceedings of the Eighteenth International Conference on Intelligent User Interfaces, ACM, New York, NY, USA, pp. 317–328.

Tümen, R.S., Acer, M.E., Sezgin, T.M. , 2010. Feature extraction and classifier combination for image-based sketch recognition. In: Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium, Eurographics Association, Aire-la-Ville, Switzerland, pp. 63–70.

Yi, W., Ballard, D.H., 2009. Recognizing behavior in hand-eye coordination patterns. Int. J. Hum. Robot. 6 (3), 337–359.

Yu, C., Ballard, D., 2002. Learning to recognize human action sequences. In: Proceeding of the Second International Conference on Development and Learning, pp. 28–33.

Yu, C., Ballard, D.H., 2002. Understanding human behaviors based on eye-head-hand coordination. In: Proceedings of the Second International Workshop on Biologically Motivated Computer Vision, Springer-Verlag, London, UK, pp. 611–619.

Zhai, S., Morimoto, C., Ihde, S., 1999. Manual and gaze input cascaded (magic) pointing. In: Proceedings of the SIGCHI conference on Human Factors in Computing Systems, ACM, New York, NY, USA, pp. 246–253.