Improving Finger Stroke Recognition Rate for Eyes-Free Mid-Air Typing in VR

Yatharth Singhal
The University of Texas at Dallas
Richardson, TX, USA
Yatharth.Singhal@UTDallas.edu

Richard Noeske
The University of Texas at Dallas
Richardson, TX, USA
Richard.Noeske@UTDallas.edu

Ayush Bhardwaj
The University of Texas at Dallas
Richardson, TX, USA
Ayush.Bhardwaj@UTDallas.edu

Jin Ryong Kim
The University of Texas at Dallas
Richardson, TX, USA
Jin.Kim@UTDallas.edu

ABSTRACT

We examine mid-air typing data collected from touch typists to evaluate the features and classification models for recognizing finger stroke. A large number of finger movement traces have been collected using finger motion capture systems, labeled into individual finger strokes, and classified into several key features. We test finger kinematic features, including 3D position, velocity, acceleration, and temporal features, including previous fingers and keys. Based on this analysis, we assess the performance of various classifiers, including Naive Bayes, Random Forest, Support Vector Machines, and Deep Neural Networks, in terms of the accuracy for correctly classifying the keystroke. We finally incorporate a linguistic heuristic to explore the effectiveness of the character prediction model and improve the total accuracy.

CCS CONCEPTS

• **Human-centered computing** → **Text input**.

KEYWORDS

Mid-Air Typing, Eyes-free Typing, Text Entry, Deep Neural Network, Keystroke Classification

ACM Reference Format:
Yatharth Singhal, Richard Noeske, Ayush Bhardwaj, and Jin Ryong Kim. 2022. Improving Finger Stroke Recognition Rate for Eyes-Free Mid-Air Typing in VR. In *CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3491102.3502100

1 INTRODUCTION

In recent years, text entry in virtual reality (VR) has become essential as a large number of applications require a considerable amount of text input. With a variety of VR applications in business, games, social media, and entertainment, people now chat with their friends, configure application settings, and enter keywords to search online. Text entry is becoming an indispensable part of VR applications, and thus, an efficient text entry solution is needed to deliver seamless and enjoyable VR experiences.

A variety of approaches have been proposed to satisfy this need for effective text entry in VR [5]. Some solutions involve using apparatus, such as hand-held controllers [2, 19, 26, 42], touchscreen [14, 15], fingers [6, 7, 11], eyes [18], or the head [45] to point to keys on a virtual keyboard. These solutions typically struggle with a lack of efficiency and introduce a significant amount of occlusion. Other solutions use unique gestures from hands [31, 37], styluses [3, 24, 28, 32, 41] or other wearable devices [17, 29, 39] to generate text input. However, each of these methods requires learning a unique and often unfamiliar text entry method, which inhibits its usage at a large scale. The introduction of a physical keyboard into a VR scene to allow users directly access the keyboard has also been investigated, typically providing elevated typing speeds, but either restricting users to a single space [23] or requiring them to wear additional equipment [33].

Freehand typing in mid-air aims to benefit from the higher typing speeds yielded by approaches using physical keyboards. Simply typing in the air, relying on human's motor system as if typing on a physical keyboard can bring performance benefits and freedom to move around the scene. These methods make use of markers [25] or hand tracking devices [43, 44] to monitor hand motions and identify keystrokes based on these motions. Further analysis of hand motions made without the visual or physical feedback of a keyboard reveals some challenges involved in fully utilizing unconstrained freehand typing, particularly in finger stroke detection and finger and key classification of these finger strokes [12]. Finger flexion on its own is not enough to effectively identify keystrokes when engaging in unconstrained freehand typing, and the amplitude of fingers is not as effective in discriminating between different fingers/keys due to a high amplitude ratio between active and passive fingers. That is, when making a keystroke, tendons in hand cause the finger to press the key (active finger) and the other fingers (passive fingers) to make correlated movements, reducing the effectiveness of the amplitude of each finger in predicting the keystroke. This effect is most evident when users are conducting unconstrained typing, where any cues that would indicate that users have pressed a key are absent. This effect is also more prevalent when users attempt to type naturally without making clear, individuated keypresses.

CHI '22, April 29-May 5, 2022, New Orleans, LA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9157-3/22/04…$15.00
https://doi.org/10.1145/3491102.3502100

These correlated movements result in reduced finger individuation which reduces the effectiveness of amplitude in finger classification. When classifying fingers and keys, it is pivotal to examine features related to typing motions, such as positions and velocities of the fingertips, to gain a better understanding of what qualities of hand motion most effectively convey the specific keystroke and individual performs. Effective evaluation of features and models will allow finger and key classifiers to efficiently use data and more accurately differentiate between keystrokes, providing a significantly smoother typing experience to users.

In this paper, we evaluate techniques to improve finger stroke recognition for eyes-free typing in mid-air. We collect 30,901 unconstrained touch typing traces in mid-air using hand motion tracking devices and process the data to label the finger stroke. We then evaluate the performance of kinematic and temporal features that impact the recognition accuracy of identifying fingers and keys. We further assess the performance of four classifiers and explore the linguistic heuristics to enhance the recognition rate. The main contributions of this paper are 1) an analysis of features and classifiers of unconstrained typing data to achieve high accuracy in finger stroke recognition, 2) the development of a classification model that accurately recognizes the finger stroke and predicts the intended keypress based on the combination of classifier and character language model, and 3) a design of a pipeline that shows the process of building a recognizer for eyes-free typing.

2 RELATED WORK

2.1 Selection-based Text Entry

Several approaches have been proposed for text input solutions for VR, and one of the intuitive and popular methods is the selection-based approach using the visual layout of the keyboard. A selection-based text entry is based on a key selection mechanism using ray-cast pointing (or tapping) on a key on the visual layout of the keyboard to register the key. The users use their hand-held controllers [2, 19, 42], hands and fingers [6, 7, 11], or other parts of the body (e.g., head or eyes movements) [18, 45] to select the target keys. It's simple and easy to use but suffers from a lack of efficiency and occlusion problems. Speicher et al. [36] studied the design space to evaluate the selection-based text entry in VR. Their study compared head-pointing, controller pointing, controller tapping, freehand, and discrete/continuous cursor techniques and confirmed that pointing using tracked hand-held controllers outperforms all other methods. However, their study also suggested following the design guidelines based on the design considerations and given resources. While typing performance on various studies using controllers ranges from 8.59 WPM [46] through between 13.57 and 24.61 WPM [2, 19, 42], other studies using body parts showed promising performance. Dudley et al. [7] presented a fast and precise text input system using fingers and reached up to 17.75 WPM with CER of 1%. In their other studies [6, 11], they compared the performance of four text entry strategies: typing with two index fingers on a surface (55.6 WPM) and in mid-air (42.1 WPM), typing with ten fingers on a surface (51.6 WPM) and in mid-air (34.5 WPM) with a virtual keyboard, and confirmed that higher typing speed was achieved when users are engaged in surface typing. Adhikary and Vertanen [1] investigated a method of using hand tracking to type on a vertical projection of a keyboard positioned in front of the user, achieving an average of 16 WPM. Yu et al. [45] proposed a head-based text entry for HMDs to allow users to control a pointer on a virtual keyboard using head rotation. Their study showed that the typing speed could be achieved up to 24.73 WPM after 60 minutes of training with a gesture-word recognition algorithm with head movement patterns.

2.2 Mid-Air Gesture

Mid-air gesture-based text entry methods use the free motion of devices or fingers (or heads) to express words or characters for key entry in VR. These methods are originally used for stylus-based text entry to make a distinct gesture for words, such as SHARK [24], Cirrin [28], and QuickWriting [32], or for characters, such as Graffiti [3], Unistrokes [3], and EdgeWrite [41]. These methods are potentially fast and easy to learn but are still limited in typing performance. Vulture [29] is a mid-air word-gesture keyboard that uses a glove with markers to recognize the gestures. They used hand movement to recognize a word based on word-gesture algorithms and pinch gestures for a word delimiter. They reported that users could achieve up to 28.1 WPM with training. RotoSwype [17] is also a word-gesture typing technique that uses a wearable device (ring) in mid-air. Their technique is based on a ring-tilt word-gesture typing that allows one-handed typing, and it could achieve 14 WPM with 1% uncorrected error rates with training. Chen et al. [4] also explored word-gesture level text entry using 6-DOF controllers and compared it with pressure-sensitive touchscreen input. In their early stage of work, users could achieve up to 34.2 WPM with training, and it outperformed the touchscreen input (22.4 WPM with training). Shen et al. [35] investigated methods of processing gestures made by fingers sliding across the keys of a projected keyboard to type entire phrases. AirStroke [31] is a character-level gesture typing technique that is based on Graffiti [3]. It uses both hands for selection and word completion, yielding 13 WPM with word completion. Sridhar et al. [37] investigated the text input method using the free motion of multiple fingers in mid-air. Their study closely looked into several factors, including individuation, coactivation, and learnability, and reported that they achieved up to 22 WPM with optimization. The details and performance of this system are discussed in more detail in their other study [8].

2.3 Physical Keyboards Blended into the Scene

Physical keyboards are used by blending them into the VR scene. This allows users to access their keyboards directly while engaging in a virtual space, providing high speed typing with low error rates. A downside to this method is that it tethers a user to a single physical space [23]. McGill et al. [30] tackled VR typing usability issues with HMD by blending physical keyboards into the VR scene. They found that blending the physical keyboard significantly reduced the error rate. Walker et al. [38] proposed a system that provides visual feedback of automatic correction. They reported that users could reach over 40 WPM with less than a 5% error rate. Grubert et al. [14] explored the performance and user experience of a physical keyboard used in VR applications and compared it with that using touchscreen keyboards. They confirmed novice users performed at about 60% of their typing speed using the physical keyboard.

Knierim et al. [23] investigated the effects of avatar hands in typing performance. The results showed that displaying hands is beneficial to all typists, and the benefits were more evident for experienced typists. Pham et al. [33] introduced HawKEY which provides a simple and efficient text entry solution without restricting physical movement by placing the keyboard on a tray in front of the user.

2.4 Freehand Typing

Freehand typing is a text entry method that allows users to type in the air by recognizing finger strokes based on their finger movements. This technique has many benefits as it has a high bandwidth of text input and is not tethered to a physical input device. It is also natural and easy to use as users simply type in the air utilizing their knowledge of typing on physical keyboards. Numerous approaches have been presented to explore the feasibility of freehand typing. ARKB [25] is a 3D vision-based freehand typing device that relies on markers on the fingers. It provides natural interaction with fingers and audio-visual feedback through AR glasses. TiTAN [43] is a ten-finger mid-air typing system that allows users to touch-type using a Kinect v2 and a Leap Motion hand tracking device. The users can tap any finger to register a key based on the finger's current position. TiTAN achieved up to 9.4 WPM. ATK [44] is a mid-air typing system that recognizes finger tapping using Leap Motion hand-tracked data to allow ten finger freehand typing. Finger tapping recognition was based on a probabilistic tap detection algorithm and augmented version of Goodman's input correction model [13]. ATK achieved up to 29.2 WPM with training. Gil et al. [12] explored unconstrained eyes-free in-air typing of touch typists. Their goal was to characterize typing finger movements, including properties of finger kinematics, correlated movement of fingers, 3D distributions of keys, and typing strategies of in-air typing for design recommendations for mid-air freehand typing. Their study indicated that the typing speed achieved up to 49.1 WPM, showing the great potential of this approach.

3 PROPOSED PIPELINE

Figure 1 shows the proposed pipeline for finger recognition and intended keypress identification. As a first step, typing traces for touch typists are collected using the finger tracking system with our recording software. These traces are then processed for missing values which are then eliminated using the mean imputation technique. This data is then loaded into our custom simulator, which helps us label the appropriate key and finger to the data frame by allowing navigation through a recording of the hand movements. Users manually indicate in which of these frames a key became fully pressed, and the simulator stores the relevant features of the selected frames. This labeled data is then used in feature evaluation to find the highest impact features for finger and key classification. These features can be classified into two broad categories: kinematic and temporal features. We evaluate the performance of these features by passing them as a single feature vector to a generic classifier.

The features selected from the feature extraction phase are passed to four different classifiers to evaluate which machine learning model performs better with the given data and feature set. After selecting a particular model, it is trained on the dataset with features chosen to recognize the finger. The output of the finger recognizer is passed as a feature along with the features selected for training the key classification model. The classification model assigns probabilistic scores to all the keys based on the training model, and then the key with the highest probabilistic score is used as a prediction for that particular keypress. To further improve the recognition rate of the key classifier, a character language model is trained, and the weighted average of the confidence scores from both models is used to predict the keypress.

4 DATA COLLECTION

Data collection aims to evaluate and identify the features and classification models for finger stroke recognition by collecting a large number of in-air finger stroke traces. To achieve this goal, we developed a set of tools that allow recording finger stroke movements using motion tracking systems and simulating participants' hands and fingers to facilitate labeling keystrokes, described in the following subsections.

4.1 Participants

A total of twenty touch-typists (female=5; mean age=28.4; SD=7.82, two left-handed) participated in this study. Among them, sixteen typists (female=5; mean age=30.06; SD=7.76, two left-handed) conducted the typing task experiment using a Leap Motion finger tracking setup [12]. We recruited additional four typists (male = 4; mean age=26.75; SD=7.88) to perform the same experiment using the OptiTrack motion capture system, which provides more precise finger tracking data. The mean typing speed was 68.22 WPM with 1.2% of uncorrected error rate. Each participant was paid with a $10 gift card for their participation. All the experiments were approved by the Institutional Review Board (IRB) at the University of Texas at Dallas (IRB-21-194).

4.2 Apparatus

4.2.1 Finger Tracking. We used two different finger tracking devices for data collection: Leap Motion and OptiTrack. The Leap Motion tracking device is cost-effective and provides a simple setup, but tracking is less accurate than a marker-based precision tracking system. OptiTrack hand tracking provides precision finger tracking with 1.3 MP resolution, capturing precise finger movements in a sub-millimeter unit. We converted all the 3D position data collected to a coordinate system relative to the palm position to remove any discrepancies that might arise from a different frame of reference in both trackers.

Leap Motion Setup. Data collection for finger movements in [12] was conducted using a Leap Motion tracking device (version 3.2.0+45899). The tracking device was placed in front of participants, elevated 41 cm above the floor.

OptiTrack Setup. A total of six OptiTrack Prime 13 cameras were positioned around the table which participants conducted mid-air typing (see Figure 2). Markers were placed on participants' fingernails, on top of the finger joints, on the back of each hand, and above the palm.

4.2.2 Recorder. Figure 3 shows the recorder that collects finger motion data while providing a set of randomly selected phrases from Mackenzie's phrase set [27] for typing tests. The virtual scene,

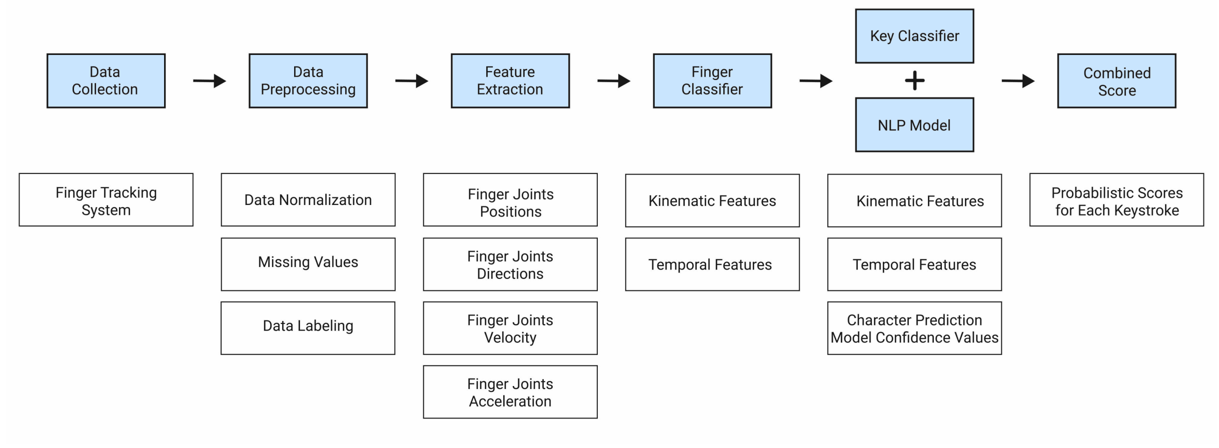

Figure 1: Pipeline of the process from raw finger tracking data to keystroke prediction.

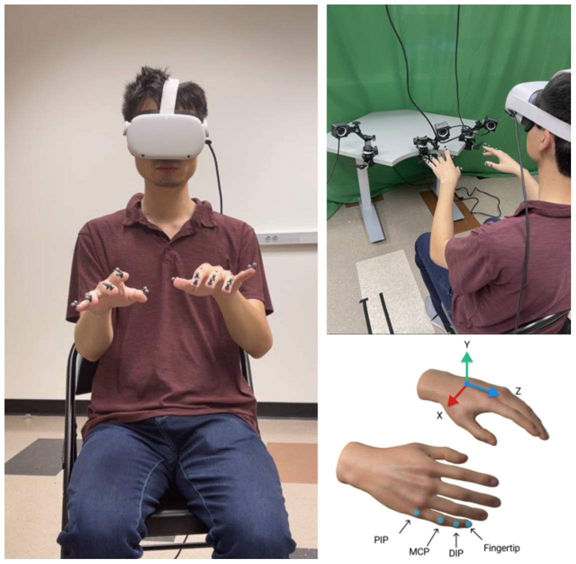

Figure 2: Experiment setup with participant in typing position (left) with OptiTrack system (top-right) and coordinate system used for the hand data along with the different finger joints tracked (bottom-right). The coordinates for each finger are stored relative to the palm of its hand.

built using Unity3D, displays two virtual hands representing the participant's tracked hand positions. It also contains two spheres to serve as a reference point for users to start typing, with the left and right spheres representing the positions of the F and J keys on a physical keyboard, respectively. These spheres were spaced 9 cm apart to ensure a comfortable spacing between the participant's hands. Participants could adjust the positions of these spheres to a location where they could type the most comfortably. The virtual scene also contained a text display area located directly in front of participants after loading the virtual scene, which displayed the phrases that participants would type along with the trial number.

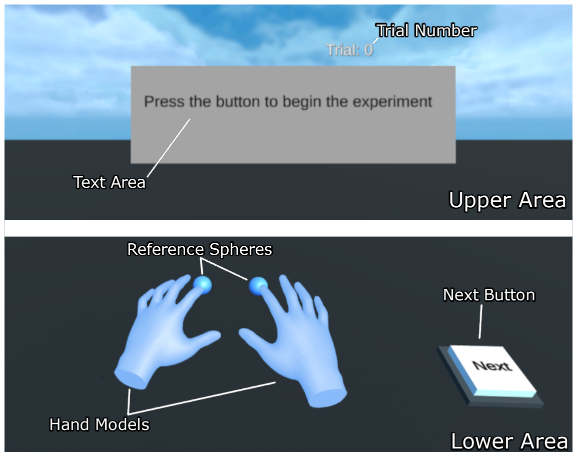

Figure 3: Participant view of the recorder. The display text is kept at eye level, while the reference spheres and the next button are lower, near the hands.

A next button was located to the right of participants, allowing them to move on to the next phrase after they finished typing the current one. We did not intentionally show any virtual keyboard in the experiment. This was done to ensure that the finger movement data was from "eyes-free" typing [12]. We deliberately removed the visual constraint of the virtual keyboard because it would be impossible to observe unconstrained typing finger movements while showing a virtual keyboard. By allowing the visual layout of the keyboard, people may look at the layout while making keystrokes that generate eye-gaze shifts between text and keyboard areas, affecting typing speed. We further argue that providing minimum visual cues is best to induce natural high-speed typing behavior. We also believe that the finger movements of "eyes-free" typing without visual constraints can further provide ground truth about users' expected locations for keys, which can be used for future virtual keyboard design. No additional visual feedback for keyboard position or keystrokes were provided to most effectively capture natural typing behavior and finger stroke movements for eyes-free typing. The recorder supports both Leap Motion and OptiTrack devices.

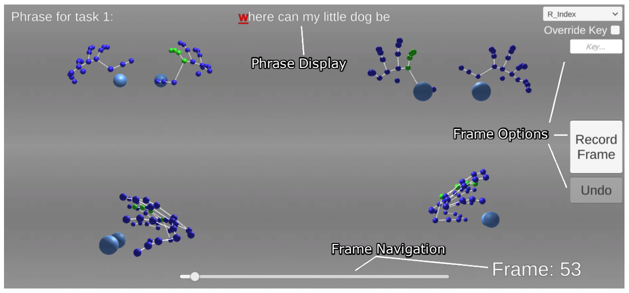

Figure 4: Overview of the simulator. Users are able to pause/play hand motions using space and use the directional keys to navigate between frames precisely.

4.2.3 Simulator. After data was collected using the recorder, it was processed in the simulator, which was also constructed using Unity3D. The simulator renders the participant's hands using the collected motion capture data to allow for frame-by-frame examination of mid-air typing. This tool was used to identify and record frames where keystrokes were executed manually. As shown in Figure 4, the simulator contains four different camera views of both hands so that key-presses could be detected from different angles. It also contains a simple user interface designed to facilitate the process of labeling keystrokes. At the top of the screen, the phrase that the participant was typing is shown, and the character for the next keystroke is highlighted. Individuals labeling the data would then navigate to the specific frame at which this keystroke was conducted and record the keystroke by pressing a record frame button to the right of the screen. Provisions were also included to override the recorded character with one outside of the phrase to account for situations where characters not included in the phrase were typed, such as situations where participants felt they made a mistake in typing and pressed the backspace key. The simulator supports datasets from both Leap Motion and OptiTrack devices.

4.3 Procedure

Instructions were given to each participant with a short description of the study, the aim, and the procedure before the main experiment. Before the main experiment, the participants were asked to type 20 phrases on a physical keyboard using TextTest [40] to verify their finger-to-key mapping and typing performance. The choice of 20 phrases was to make it equivalent to one block the participants would be typing in the actual experiment. For participants with an OptiTrack system, IR reflective markers were placed on the participants' hands by the experimenter. A marker was placed on the fingertip, endpoint, and the joints between the phalanx bones for each finger. This constituted a total of 38 markers (4 per finger and 3 per thumb). Additionally, two more markers were placed on the center of each palm. This was done to reciprocate the tracking done by the Leap Motion camera to make the data collected by two different sources compatible. This marker placement process was omitted for participants with a Leap Motion.

The participants first completed an informal training session to get used to the software. The trial began when the participants placed their virtual index fingers into the two spheres. The spheres then disappeared, and the test phrase was displayed. The experiment consisted of four blocks (one practice and three actual), with twenty phrases in each block. Participants were asked to type as fast and accurately as possible. They were encouraged to hit the backspace key to correct any errors. After finishing typing the sentence, they were instructed to hit the next button to go to the subsequent trial, which was started by touching the spheres simultaneously using two index fingers again. In order to minimize fatigue, participants were asked to take a break after each block.

4.4 Data Processing

We collected a total of 30,901 labeled finger strokes in the form of 3D positions using both Leap Motion and OptiTrack tracking systems. This data was then used to derive movement direction, velocity, and acceleration for all fingers and both hands. We parsed this raw data into our simulator to visualize the typing sequence. The simulator loads the 3D position data of all fingers' stroke movements, providing a visual environment of the participant's hands with frame-by-frame inspection for manual labeling of appropriate characters. We converted all the data into the relative coordinate frame (Figure 2) to remove any disparities resulting from the difference in two different data collection sources (Leap Motion and OptiTrack). The final data set includes information of all 3D positions, directions, velocities, and accelerations of 30,901 finger strokes for all ten fingers and two palms, at the frames where the finger making the keystroke has fully pressed the key.

5 FEATURE EVALUATION

Unconstrained typing movements are complex. Due to the composition of muscles and tendons in human hands and fingers, the intended movement of one finger can yield correlated movements in other fingers. This higher correlation of finger movements in unconstrained typing can lead to a higher false detection ratio in finger stroke recognition [12]. This section evaluates a set of features for finger and key classification with unconstrained typing movements. We typically assess the performance of features that are relevant to correlated finger movements in both active fingers (intended movements for key entry) and passive fingers (unintended but correlated movements), as well as temporal features that consider previous fingers and keys.

5.1 Features

Two types of features are evaluated: kinematic features and temporal features. Kinematic features are features related to finger movements in both active and passive fingers. Temporal features are time-related features that consider linguistic information.

5.1.1 Kinematic Features.

Positions. We considered the X, Y, and Z position of every fingertip, joint, and endpoint for each finger and palm. They were then converted into relative positions, with the pertaining palm position being the point of reference.

Movement Velocity. We calculated the velocity by taking the 3D vector of the position at the current frame and subtracting that of the previous frame from it. This gave us a vector indicating the distance traveled, and the magnitude of the vector, divided by frame rate, was recorded as the velocity of the tracking point.

Movement Direction. Using the calculation to measure distance traveled between frames, we normalized the provided distance vector to obtain a unit vector indicating the tracking point's direction.

Acceleration Magnitude. The acceleration of tracking points was calculated by taking the current frame's velocity vector and subtracting the previous frame's velocity vector from it. The magnitude of the resultant vector, divided by the frame rate to account for the length of a frame as a sixtieth of a second, was recorded as the acceleration magnitude.

5.1.2 *Temporal Features.*

Previous Finger. The previous finger is an active finger that performs the keypress. It was set to null for the first keypress within a phrase.

Previous Key. The previous key is a linguistic feature that records the previous character entered. It was set to null if the previous key press didn't exist.

5.2 Feature Performance

The performance of each feature is tested by running a *RandomForest* classifier and passing only that feature as a feature vector. The model is trained and tested using 10 fold cross-validation, and we report the accuracy, F1 score, precision, recall, and area under curve (AUC) for kinematic features and temporal features.

5.2.1 Fingers. Table 1 shows the performance of kinematic features on finger classification. We observed that the position and direction had a significant impact while the velocity and acceleration did not. This is probably because the changes in 3D position and direction of the active finger are significantly larger than that of other passive fingers.

We further analyzed which joints have better performance by selecting positions and directions as relevant features. We performed feature evaluation on *Fingertip*, *DIP* (Distal Interphalangeal), *MCP* (Metacarpophalangeal), and *PIP* (Proximal Interphalangeal) (see Figure 2) by feeding them as a feature into a *RandomForest* classifier and found that *Fingertip* (accuracy=0.909), *DIP* (accuracy=0.913), and *MCP* (accuracy=0.898) had significantly higher accuracy ratio than *PIP* (accuracy=0.621). This is probably due to the fact that *PIP* is close to the palm and therefore does not have as much of an effect on the changes in position.

Table 2 shows the performance of temporal features on finger classification. We confirmed that including temporal features can improve the classification rate. Among them, providing both the previous finger and previous key showed the best performance.

5.2.2 Keys. Table 3 shows the performance of kinematic features on key classification. The *RandomForest* classifier was used and compared the features based on accuracy, F1 score, precision, recall, and area under curve (AUC). Similar to finger classification, we confirmed that the accuracy of positions and directions is higher than the accuracy of velocity and acceleration. Still, their ratios in key classification were relatively lower than those in finger classification.

Table 4 shows the performance of temporal features on key classification. Providing temporal features showed higher accuracy, and the best performance was achieved when both the previous finger and previous key were used as features. The overall performance of temporal and kinematic features in terms of accuracy, precision, and recall on key classification was relatively lower than the performance of those in finger classification.

Table 1: Performance of Kinematic Features on Finger Classification

FEATURE	ACCURACY	F1 SCORE	PRECISION	RECALL	AUC
Position	0.910 ± 0.012	0.909	0.908	0.908	0.991
Velocity	0.417 ± 0.016	0.391	0.423	0.392	0.811
Direction	0.911 ± 0.014	0.914	0.915	0.916	0.992
Acceleration	0.543 ± 0.012	0.533	0.547	0.545	0.887

Table 2: Performance of Temporal Features on Finger Classification

FEATURE	ACCURACY	F1 SCORE	PRECISION	RECALL	AUC
None	0.912 ± 0.015	0.913	0.914	0.914	0.991
Previous Finger	0.922 ± 0.019	0.923	0.922	0.922	0.992
Previous Key	0.923 ± 0.024	0.923	0.924	0.924	0.993
Both	0.929 ± 0.018	0.926	0.927	0.927	0.993

Table 3: Performance of Kinematic Features on Key Classification

FEATURE	ACCURACY	F1 SCORE	PRECISION	RECALL	AUC
Position	0.863± 0.023	0.859	0.863	0.862	0.996
Velocity	0.619± 0.015	0.572	0.605	0.619	0.972
Direction	0.869 ± 0.020	0.862	0.864	0.864	0.995
Acceleration	0.605 ± 0.026	0.553	0.581	0.602	0.967

Table 4: Performance of Temporal Features on Key Classification

FEATURE	ACCURACY	F1 SCORE	PRECISION	RECALL	AUC
None	0.864± 0.011	0.854	0.862	0.864	0.994
Previous Finger	0.881± 0.021	0.879	0.883	0.880	0.996
Previous Key	0.888 ± 0.015	0.889	0.888	0.922	0.996
Both	0.897± 0.019	0.897	0.899	0.894	0.996

6 RECOGNITION MODEL

6.1 Model Selection

Four machine learning classifiers were selected and compared to identify which model worked the best on the data. These were the *Naive Bayes*, *Support Vector Machine (SVM)*, *RandomForest*, and *Deep Neural Network*. Each model was trained on 80% of the data, and the remaining 20% was kept for validation purposes. The classifier only reads a single frame of data to make a prediction, particularly the frame where the finger pressing the key has reached its maximum point of flexion. The description of each classifier is as follows:

6.1.1 Naive Bayes. Naive Bayes is a probabilistic model, which is one of the simplest yet most commonly used classifiers. The model calculates the probability of a finger being the active finger and the key being pressed given the selected features using Bayes Theorem.

6.1.2 Support Vector Machine (SVM). SVM is a support vector machine with a linear kernel that was trained for classification. This algorithm creates hyper planes to classify data into various classes. SVMs are used to solve many real-world classification problems and can act as a reliable baseline.

6.1.3 Random Forest. Random Forest is an ensemble, learning-based classification technique that uses a myriad of decision trees during the training process. The selected features were passed as a feature vector, and the output is the class selected by the highest number of trees in the ensemble.

6.1.4 Deep Neural Network. The deep neural network is typically suited for dealing with large data sets due to its complex architectures containing multiple hidden layers and many neurons. In this study, the architecture chosen for the neural network comprises of the following layers: 512 nodes dense layer with batch normalization with ReLU activation, followed by another 256 nodes dense layer with batch normalization, ReLU activation, and 30% dropout, then a 128 node dense layer with batch normalization, ReLU activation and 30% dropout, a 64 node dense layer with batch normalization with ReLU activation and finally a ten node dense layer for finger classification or a 28 node dense layer for key classification with a sigmoid activation function. The network, therefore, has approximately 350,00 trainable parameters. A sparse categorical cross-entropy loss function is used because the classes for this problem are mutually exclusive. The cross-entropy loss function is a probabilistic loss that is minimized when the model's predictions become similar to the ground truth [16]. The hyperparameters were computed using Grid Search. We trained the algorithm for all combinations by using the two sets of hyperparameters (learning rate and number of layers) and measured the performance using the *Cross Validation* technique.

6.2 Model Comparison

Table 5 shows the performance metric comparison among all four models for finger classification. *Naive Bayes* was found to be the worst performing model with an accuracy of about 73%. *SVM*, *RandomForest*, and the *Neural Network* performed adequately, but the *Deep Neural Network* model outperforms others with an overall finger classification accuracy of approximately 96%.

Table 6 shows the performance metric comparison among the four models for key classification. Again, the *Deep Neural Network* model outperforms others with an overall key classification accuracy of 91.7%. *RandomForest* (accuracy = 89.5%) classifier performs better than *SVM* (85.2%) and *Naive Bayes* (57.5%) was the worst performer. We observe that the key classification yields slightly lower accuracy than the finger classification; this is probably due to fewer classes in the finger classification.

Table 5: Performance of Models on Finger Classification

MODEL	ACCURACY	F1 SCORE	PRECISION	RECALL	AUC
Naive Bayes	0.736 ± 0.009	0.734	0.759	0.727	0.944
SVM	0.917 ± 0.021	0.899	0.909	0.895	0.993
Random Forest	0.926 ± 0.018	0.928	0.928	0.928	0.993
Deep Neural Net	0.954 ± 0.013	0.947	0.943	0.942	0.996

Table 6: Performance of Models on Key Classification

FEATURE	ACCURACY	F1 SCORE	PRECISION	RECALL	AUC
Naive Bayes	0.575 ± 0.011	0.586	0.644	0.577	0.951
SVM	0.852 ± 0.024	0.793	0.811	0.779	0.993
Random Forest	0.895 ± 0.015	0.892	0.896	0.893	0.996
Deep Neural Net	0.917 ± 0.017	0.911	0.915	0.913	0.996

6.3 Character Language Model

We used a Recurrent Neural Network (RNN) for the character language model. The model consists of Character Embedding, Gated Recurrent Units (GRU) Layer, Dense Layer, and Logits as an output layer. The model looks up the embedding for each character, runs the GRU one timestep with the embedding as input, and applies the dense layer to generate logits predicting the likelihood of the next character. We used Wikicorpus [34], an open-source corpus containing around 600 million words, to train the model. We only used the lowercase alphabets.

6.4 Results and Analysis

After integrating the character language model with the key classifier, we improved the accuracy by 2.4%, yielding a total accuracy of 94.1%. This clearly shows that adding linguistic features in mid-air typing helps recognize the intended keypress.

We assigned different weights to each model's output to get the highest accuracy for keystroke classification. The evaluation was done with various combinations, and we found that a 70:30 ratio of weights for the keystroke and character language models produced the most desirable results. Note that we did not use the output of the natural language processing (NLP) model until the typed sentence length was more than one character as it gives a very high variance among its confidence values if otherwise. The NLP model character predictions were more accurate for the last character prediction of each word in the sentences.

7 DISCUSSION

This study investigated the performance of features and classifiers on finger stroke recognition for eyes-free mid-air typing in VR. We showed an approach to achieve unconstrained freehand typing in mid-air by evaluating kinematic and temporal features and different classifiers with a character language model. Unconstrained typing finger movements were captured and processed for feature and model selection. We showed that the kinematic features of position and direction are critical and that temporal features can further improve accuracy. We further showed that the Deep Neural Network model outperforms other finger and keystroke recognition classifiers. The character language model was also significant in improving the accuracy of the keystroke classifier. Overall, we showed that the performance of those features and classifiers demonstrated the great potential of this approach and possibilities.

One of the significant contributions in this work is a machine learning-based analysis of features and analysis of unconstrained tying data. Unconstrained typing [12] involves faster, shorter, and more interleaved and inter-correlated finger motions than constrained typing [43, 44]. Unlike constrained typing, where it only considers the amplitude of the fingers, unconstrained typing requires multiple factors because of the high amplitude ratio between active and passive fingers. In fact, we found that features related to the form of the hand (i.e., position and direction) showed a higher accuracy compared to those related to the movement of the hand (i.e., velocity and acceleration), indicating that features related to the overall form of the hand can be significant features. In unconstrained typing, making a clear tap of single finger stroking is challenging because the physical and neuromuscular traits of the hand make it difficult to move a single finger without moving the others [11]. We also found that temporal features, such as the previous finger used and previous key used, showed accuracy improvements, demonstrating the feasibility of utilizing temporal features.

Another significant contribution is the development of classifiers. We tested four different model architectures: Naive Bayes, Support Vector Machine, Random Forest, and Deep Neural Network, and found that the Deep Neural Network outperforms others with an accuracy of 91.7%. We believe that it is because the feature-set being used for classification is complex, and the Deep Neural Network performed better in processing complex data than other techniques. We further reported an increase of 2.4% in classification accuracy with integrating a character language model, demonstrating a great candidate for enhancing the recognition performance. This shows that contextual information is essential in classifying the keystroke.

We also proposed a pipeline showing an entire process to build a recognizer for unconstrained finger typing in mid-air. Active finger information from the finger classifier and the kinematic and temporal features is passed into a key classifier that assigns probabilistic scores to all the keys based on the training model. Then the key with the highest probabilistic score is used to predict a particular key press. We further introduced software tools to support the process effectively. We developed software that supports the partially-automated labeling process. The software automatically reads the phrases and suggests the target finger based on the finger-to-key mapping. The labeled keystrokes are further processed, and various features are tested to find the features necessary for finger and key classification.

One of the limitations of this study is that it examined only the typing patterns of touch typists. Due to this, there was a strict finger-key mapping according to the conventions of touch typing. Keystrokes that did not follow the typical finger-key mapping were ignored to ensure the models were only provided with accurate touch typing data. We recruited touch typists because we believe they are a critical user group to design the most efficient text entry system [10], and studying them will eventually lead to a design guideline for novice users [9, 20–22], which we plan to explore in the next phase with different types of typists.

Currently, our recorder only supports OptiTrack and Leap Motion hand tracking. It would greatly benefit from being able to support other tracking devices to expand its range of use and would also benefit from being better equipped to handle the different coordinate systems used by different tracking devices. Adding feedback to keystrokes in the Recorder, such as a clicking sound and the appearance of an asterisk when a key is pressed may provide more realistic typing data, as it is extremely likely that implementations of freehand typing in VR applications would come with such feedback, and could potentially effect how individuals type in mid-air.

We have a number of clear action items to extend this study. Ultimately, the classifier created in this paper should be able to classify fingers and keys in real-time. To achieve this, a method to actively detect finger strokes in mid-air through hand tracking devices should be proposed. This recognizer would have to be able to identify the furthest point of flexion of the keystroke so that the classifier can collect the relevant hand data and classify the keystroke.

We also have a plan to improve the data processing pipeline by implementing the automatic labeling of data. A system that could accurately identify and record the frames, or assist in navigating to the frames of potential keystrokes would greatly expedite the processing of data. This would make the process of gathering a large amount of data to improve the existing models or create new models much more feasible.

8 CONCLUSION

In this paper, we presented an empirical study to evaluate the features and classifiers and improve the finger stroke recognition for mid-air typing. We tested a set of kinematic and temporal features retrieved from the dataset. We further compared four machine learning classifiers, including Naive Bayes, Support Vector Machine, RandomForest, and Deep Neural Network. We also examined the effect of the character language model. Finally, we closed by discussing insights from this study, how to improve finger stroke recognition, and the future work to expand this study.

ACKNOWLEDGMENTS

We would like to thank Haokun Wang and Henry Kim for their efforts in data collection and annotation process. Supporting data for this publication is available at: https://mi-lab.io/data/in-air-keyboard-user-data.zip.

REFERENCES

[1] Jiban Adhikary and Keith Vertanen. 2021. Typing on Midair Virtual Keyboards: Exploring Visual Designs and Interaction Styles. In *IFIP Conference on Human-Computer Interaction*. Springer, 132–151.

[2] Costas Boletsis and Stian Kongsvik. 2019. Text input in virtual reality: A preliminary evaluation of the drum-like vr keyboard. *Technologies* 7, 2 (2019), 31.

[3] Steven J Castellucci and I Scott MacKenzie. 2008. Graffiti vs. unistrokes: an empirical comparison. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 305–308.

[4] Sibo Chen, Junce Wang, Santiago Guerra, Neha Mittal, and Soravis Prakkamakul. 2019. Exploring word-gesture text entry techniques in virtual reality. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–6.

[5] Tafadzwa Joseph Dube and Ahmed Sabbir Arif. 2019. Text entry in virtual reality: A comprehensive review of the literature. In *International Conference on Human-Computer Interaction*. Springer, 419–437.

[6] John Dudley, Hrvoje Benko, Daniel Wigdor, and Per Ola Kristensson. 2019. Performance envelopes of virtual keyboard text input strategies in virtual reality. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 289–300.

[7] John J Dudley, Keith Vertanen, and Per Ola Kristensson. 2018. Fast and precise touch-based text entry for head-mounted augmented reality with variable occlusion. *ACM Transactions on Computer-Human Interaction (TOCHI)* 25, 6 (2018), 1–40.

[8] Anna Maria Feit, Srinath Sridhar, Christian Theobalt, and Antti Oulasvirta. 2015. Investigating multi-finger gestures for mid-air text entry. *ACM womENcourage* (2015).

[9] Anna Maria Feit, Daryl Weir, and Antti Oulasvirta. 2016. How we type: Movement strategies and performance in everyday typing. In *Proceedings of the 2016 chi conference on human factors in computing systems*. 4262–4273.

[10] Leah Findlater, Jacob O Wobbrock, and Daniel Wigdor. 2011. Typing on flat glass: examining ten-finger expert typing patterns on touch surfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 2453–2462.

[11] Conor R Foy, John J Dudley, Aakar Gupta, Hrvoje Benko, and Per Ola Kristensson. 2021. Understanding, Detecting and Mitigating the Effects of Coactivations in Ten-Finger Mid-Air Typing in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–11.

[12] Hyunjae Gil, Yonghwan Shin, Hyungki Son, Inwook Hwang, Ian Oakley, and Jin Ryong Kim. 2020. Characterizing In-Air Eyes-Free Typing Movements in VR. In *26th ACM Symposium on Virtual Reality Software and Technology*. 1–10.

[13] Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. 2002. Language modeling for soft keyboards. In *Proceedings of the 7th international conference on Intelligent user interfaces*. 194–195.

[14] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. 2018. Text entry in immersive head-mounted display-based virtual reality using standard keyboards. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 159–166.

[15] Jan Gugenheimer, David Dobbelstein, Christian Winkler, Gabriel Haas, and Enrico Rukzio. 2016. FaceTouch: Touch interaction for mobile virtual reality. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. 3679–3682.

[16] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International Conference on Machine Learning*. PMLR, 1321–1330.

[17] Aakar Gupta, Cheng Ji, Hui-Shyong Yeo, Aaron Quigley, and Daniel Vogel. 2019. Rotoswype: Word-gesture typing using a ring. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[18] Ramin Hedeshy, Chandan Kumar, Raphael Menges, and Steffen Staab. 2021. Hummer: Text Entry by Gaze and Hum. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–11.

[19] Haiyan Jiang and Dongdong Weng. 2020. HiPad: Text entry for head-mounted displays using circular touchpad. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 692–703.

[20] Jin Ryong Kim and Hong Z Tan. 2014. Haptic feedback intensity affects touch typing performance on a flat keyboard. In *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*. Springer, 369–375.

[21] Jin Ryong Kim and Hong Z Tan. 2014. A study of touch typing performance with keyclick feedback. In *2014 IEEE Haptics Symposium (HAPTICS)*. IEEE, 227–233.

[22] Jin Ryong Kim and Hong Z Tan. 2015. Effect of information content in sensory feedback on typing performance using a flat keyboard. In *2015 IEEE World Haptics Conference (WHC)*. IEEE, 228–234.

[23] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. 2018. Physical keyboards in virtual reality: Analysis of typing performance and effects of avatar hands. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–9.

[24] Per-Ola Kristensson and Shumin Zhai. 2004. SHARK2: a large vocabulary shorthand writing system for pen-based computers. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*. 43–52.

[25] Minkyung Lee, Woontack Woo, et al. 2003. ARKB: 3D vision-based Augmented Reality Keyboard.. In *ICAT*.

[26] Yongjae Lee and Gerard J Kim. 2017. Vitty: Virtual touch typing interface with added finger buttons. In *International Conference on Virtual, Augmented and Mixed Reality*. Springer, 111–119.

[27] I Scott MacKenzie and R William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI'03 extended abstracts on Human factors in computing systems*. 754–755.

[28] Jennifer Mankoff and Gregory D Abowd. 1998. Cirrin: A word-level unistroke keyboard for pen input. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*. 213–214.

[29] Anders Markussen, Mikkel Rønne Jakobsen, and Kasper Hornbæk. 2014. Vulture: a mid-air word-gesture keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1073–1082.

[30] Mark McGill, Daniel Boland, Roderick Murray-Smith, and Stephen Brewster. 2015. A dose of reality: Overcoming usability challenges in vr head-mounted displays. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 2143–2152.

[31] Tao Ni, Doug Bowman, and Chris North. 2011. AirStroke: bringing unistroke text entry to freehand gesture interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2473–2476.

[32] Ken Perlin. 1998. Quikwriting: continuous stylus-based text entry. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*. 215–216.

[33] Duc-Minh Pham and Wolfgang Stuerzlinger. 2019. Hawkey: Efficient and versatile text entry for virtual reality. In *25th ACM Symposium on Virtual Reality Software and Technology*. 1–11.

[34] Samuel Reese, Gemma Boleda, Montse Cuadros, Lluís Padró, and German Rigau. 2010. Wikicorpus: A word-sense disambiguated multilingual wikipedia corpus. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

[35] Junxiao Shen, John Dudley, and Per Ola Kristensson. 2021. Simulating Realistic Human Motion Trajectories of Mid-Air Gesture Typing. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 393–402.

[36] Marco Speicher, Anna Maria Feit, Pascal Ziegler, and Antonio Krüger. 2018. Selection-based text entry in virtual reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.

[37] Srinath Sridhar, Anna Maria Feit, Christian Theobalt, and Antti Oulasvirta. 2015. Investigating the dexterity of multi-finger input for mid-air text entry. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 3643–3652.

[38] James Walker, Bochao Li, Keith Vertanen, and Scott Kuhl. 2017. Efficient typing on a visually occluded physical keyboard. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 5457–5461.

[39] Eric Whitmire, Mohit Jain, Divye Jain, Greg Nelson, Ravi Karkar, Shwetak Patel, and Mayank Goel. 2017. Digitouch: Reconfigurable thumb-to-finger input and text entry on head-mounted displays. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (2017), 1–21.

[40] Jacob O Wobbrock and Brad A Myers. 2006. Analyzing the input stream for character-level errors in unconstrained text entry evaluations. *ACM Transactions on Computer-Human Interaction (TOCHI)* 13, 4 (2006), 458–489.

[41] Jacob O Wobbrock, Brad A Myers, and Duen Horng Chau. 2006. In-stroke word completion. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*. 333–336.

[42] Naoki Yanagihara, Buntarou Shizuki, and Shin Takahashi. 2019. Text Entry Method for Immersive Virtual Environments Using Curved Keyboard. In *25th ACM Symposium on Virtual Reality Software and Technology*. 1–2.

[43] Hui-Shyong Yeo, Xiao-Shen Phang, Taejin Ha, Woontack Woo, and Aaron Quigley. 2017. TiTAN: exploring midair text entry using freehand input. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. 3041–3049.

[44] Xin Yi, Chun Yu, Mingrui Zhang, Sida Gao, Ke Sun, and Yuanchun Shi. 2015. ATK: Enabling ten-finger freehand typing in air based on 3d hand tracking data. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. 539–548.

[45] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. 2017. Tap, dwell or gesture? Exploring head-based text entry techniques for HMDs. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 4479–4488.

[46] Difeng Yu, Kaixuan Fan, Heng Zhang, Diego Monteiro, Wenge Xu, and Hai-Ning Liang. 2018. PizzaText: Text entry for virtual reality systems using dual thumbsticks. *IEEE transactions on visualization and computer graphics* 24, 11 (2018), 2927–2935.