Two Hands, Two Touchpads: Optimizing Text Entry on Televisions

Brian Pham*
The University of Texas at Dallas
Richardson, Texas, USA
bap200003@utdallas.edu

Camden Thomson*
The University of Texas at Dallas
Richardson, Texas, USA
cst200000@utdallas.edu

Kuei-Yu Tsai*
The University of Texas at Dallas
Richardson, Texas, USA
kxt230002@utdallas.edu

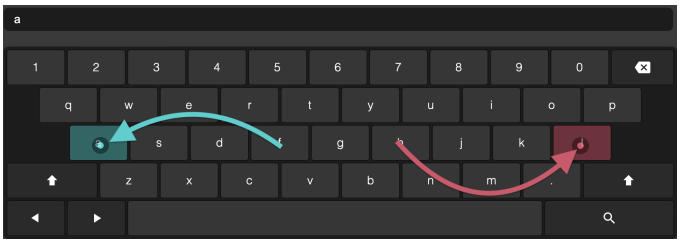

Figure 1: Text entry using dual-point touch input with absolute-positioning

ABSTRACT

The growing popularity of smart TVs has revolutionized the television viewing experience. In contrast to conventional TVs reliant on numerical channel inputs, smart TVs utilize keyword searches for enhanced user interaction. Despite this advancement, the conventional TV remote, employing directional buttons for character selection on virtual keyboards, can be cumbersome, requiring additional time and effort for typing. Furthermore, using a standard keyboard can be impractical for relaxed sofa viewing. To address these challenges, we propose a novel approach: a compact touchpad integrated into the TV remote, allowing users to intuitively swipe and click for effortless input. To validate this concept, we leverage smartphones as remote touchpads, exploring various input gestures and implementing keyword auto-suggestion to assess their usability and efficiency. Our experimentation involves statistical analysis to draw meaningful insights into the feasibility and user-friendliness of this innovative remote control design.

CCS CONCEPTS

• **Human-centered computing** → *Keyboards*; *Touch screens*; **User studies**; **Usability testing**; **Text input**.

KEYWORDS

Multi-Point Touch Input, Text Suggestions, User Experience, Usability

*All authors contributed equally to this research.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Brian Pham, Camden Thomson, and Kuei-Yu Tsai. 2023. Two Hands, Two Touchpads: Optimizing Text Entry on Televisions. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

For years, text entry has been a field necessitating fast and efficient text and character input with little to no training time. By far the most widespread solution to this problem has been the QWERTY keyboard, which has been around for over a century. Its standardized adoption and widespread familiarity have been instrumental in reducing training time, and its design is optimized for quick text entry.

With the popularization of smart devices such as smart TVs, however, the typical finger-stroke input method is not feasible given the need for a more lightweight and portable text entry solution. In the case of smart TVs, a standardized approach has been adopted using a remote with arrow keys to navigate an on-screen keyboard and select characters. These solutions, although quick to learn, lose a lot of the speed and efficiency of the more established keyboard input methods. This inefficiency can be explained by the lengthy string of key presses required for each character rather than a single keystroke.

As smart TVs continue to become more and more commonplace, the need for a more efficient text entry solution becomes imperative to improving the overall user experience. In designing such a solution, a balance must be struck between learnability and overall system efficiency. As such, it is greatly beneficial to analyze existing proven systems that users have familiarity with when designing a new and efficient text entry solution.

One such proven system is mobile touch-based keyboards, which utilize the same standardized QWERTY keyboard displayed on-screen and leverage easy and intuitive touch-based interactions to select keys. This same approach can be adapted to a smart TV environment by using a touch-based controller to control a soft keyboard displayed on the screen. Several approaches such as [1, 2] leverage this concept using various touch input methods to

control an on-screen keyboard. Most of these systems, however, use gestural interactions which for the novice user can be daunting and confusing to use. Therefore, our research aims to explore what effects touch-based input methods and auto-complete suggestions have on typing speed, accuracy, and usability when searching for content on a TV.

To solve this, we propose using simple drag-based touch interactions to select characters on the keyboard, reducing the learning curve while providing faster text input than standard remote-based input. In this proposed solution, the absolute position of the users' fingers is used to control the positioning of an on-screen cursor. The release of the finger triggers the selection of whichever key is currently highlighted, which is designed to reduce the number of required movements. This allows users to select each key in only one fluent movement to improve usability and efficiency. Additionally, we propose using two points of contact, one for each thumb, to capitalize on users' familiarity with standard QWERTY keyboards and using two hands to type.

Current commercially available systems for smart TV text input lack speed and efficiency, demonstrating the need for new efficient methods for TV-based text entry. We start section 2 by presenting related work in the domain of text entry and touch interactions. This is followed by an explanation of our proposed solution including design decisions and comparisons of different design variations. Furthermore, we examine the text-entry experience when using this system by asking participants to perform common text-entry interactions. Our findings indicate that drag-based interactions are a viable solution for smart TV text-entry, although further research and improvements are needed for it to prove more efficient and usable than existing remote-based text-entry techniques.

2 RELATED WORK

In this section, we will examine research related to alternative text input methods for television and explore current real-world practices of smartphone-TV interactions while analyzing the design considerations associated with each mode of input.

2.1 Alternative Text Input Methods

Numerous techniques for text input in TV applications already exist, encompassing a range of input modalities such as speech and gestural interactions. In this section, we will briefly discuss joysticks, speech recognition, and gestures as examples of these methods.

2.1.1 Joystick Interactions. Joystick input provides a tactile and precise method of interaction and offers intuitive navigation and accessibility, however, it tends to be slower than touch-based interactions. To increase the efficiency of joystick text input, Gu et al. [3] used joysticks to capture users' freehand writing gestures. Their method proved to be easier to learn and faster than the key selection method typically implemented with joysticks.

In another study, Go and colleagues [4] experimented with IToNe which leverages the capabilities of dual joysticks as a means of input for Japanese text. IToNe divided the Japanese syllabary chart, allocating the left half to the left thumbstick and the right half to the right thumbstick. A similar experiment by Wilson and Agrawala [5] also uses dual joysticks, but in the context of the English language and the QWERTY keyboard. During both experiments, participants rapidly input phrases while striving for maximum precision. In the trial by Go et al. [4], they found that IToNe didn't yield faster text entrty compared to a similar tool, EGCONVERT. However, they discovered that IToNe's bimanual text entry approach could be learned easily and had the potential to offer improved accuracy and smoother input experiences. The study by Wilson and Agrawala [5] also discovered that experienced users of traditional keyboards found their typing skills transferring over to their new input tool. They concluded that regardless of typing background, the tool was easily learnable and advantageous compared to single-stick selection systems.

2.1.2 Speech Recognition Interactions. In addition to the more traditional approaches discussed, researchers have also explored innovative input solutions for TVs leveraging natural language processing. For example, Hoste and Signer [6] developed a groundbreaking interface called SpeeG2, which harnesses the power of speech recognition and gesture tracking, enabling interaction with digital devices without the need for physical controllers. They accomplished this by using speech as the main input modality, supported by hand gestures which are used to confirm and correct the generated text. SpeeG2 offers the advantages of convenience and accessibility, and its applications span various domains where efficient hands-free interactions are essential.

To alleviate the challenges associated with using a remote control for text input, Wittenburg and colleagues [7] conducted a study exploring the incorporation of a microphone into the remote control. Users could initiate text input using speech by pressing a designated button on the remote. Notably, their system was designed without strict rules, allowing for unrestricted vocabulary and grammar. The researchers observed that when this speech-based input method functioned smoothly, participants found it enjoyable due to its speed and convenience. However, when the system encountered difficulties or failed to find the desired information, it led to frustration among users. This frustration primarily stemmed from the inability to provide meaningful feedback or understand why the query had failed, highlighting a notable drawback of the system.

2.1.3 Gestural Interactions. To address the preference of many users to focus more on the TV screen rather than unfamiliar controllers, Lu et al. [8] introduced BlindType. This solution leverages thumb muscle memory to facilitate efficient and eyes-free text entry using touchscreen handheld devices and lets users type on an invisible keyboard using muscle memory to remember the position of each key. BlindType delivers an adaptable and accessible typing experience, making it a valuable tool for a diverse range of users, including those with visual impairments.

As the adoption of smartwatches continues to rise among an expanding user base, Yýldýran et al. [9] designed AcousticType, a text entry method designed to meet the demands of the IoT era. AcousticType uses watch's built-in microphone to listen to a physical keyboard's acoustics and determine which key was pressed. It is then capable of sending this information to any connected IoT device, effectively turning a standard keyboard into a universal keyboard. This reduces users' reliance on screens, providing a convenient and versatile solution for text input across a wide range of contexts and devices.

In a separate study conducted by Castellucci and MacKenzie [10], they introduced UniGest a tool designed to explore using the motion-sensing accelerometer within a Nintendo Wii Remote to track user input gestures. One significant advantage of this input method was that it eliminated the need for displaying an on-screen keyboard. A recurring concern associated with on-screen keyboards is their tendency to consume valuable screen real estate and obscure content. For their solution, the researchers created a gesture alphabet chart drawing inspirations from Unistrokes, a stylus-based gesture chart, introduced by Goldberg and Richardson [11]. Both gesture charts sought to minimize the number of sweeping motions the user had to perform by designing gestures based on the frequency of character usage. On average, common alphabet characters required fewer motions while less common letters required more intricate motions. Castellucci and MacKenzie's findings indicated that UniGest had a typing speed of 27.9 words per minute, showing significant promise when compared to dual joystick text input methods.

2.2 Touch-Based Interactions

Touch interactions were popularized with the modern smartphone and have started to be adapted for use in smart TV applications. Studying current systems can provide insight into touch-based input design considerations and the effectiveness of their implementations. In this section, we will discuss the design considerations for touch-based input systems, look at other touch-based text input methods, and explore current practices regarding smartphone-TV interactions.

2.2.1 Touch Interaction Design Considerations. When designing novel touch interaction systems, there are 3 primary strategies for handling the touch input: *land-on*, which uses only the initial point of contact on the touch screen, *first-contact*, which selects the first select-able item the user comes into contact with, and *take-off*, which uses the last point of contact before a user removes their finger from the screen [12]. Each of these systems has different trade-offs, however, the *take-off* strategy consistently has fewer errors and allows users to see where their selection is before making it.

In addition to the different strategies for handling touch input, there are also different types of gestures that users can use to interact with a touch device including taps, swipes, long holds, and physical device manipulations such as tilt and shake commands. In a study by Sun et al. [13], researchers found that participants were familiar with double tapping, long holds, and horizontal scroll gestures when manipulating a video, but found vertical swiping actions hard to complete and less intuitive, suggesting that compound-movement interactions are less efficient than simple interactions.

2.2.2 Touch-Based Text Input Methods. Touch-based interactions are designed to provide a more interactive and user-friendly experience than the d-pad on a traditional remote control. In an early implementation by Choi and Li [1], an optical touchpad tracked the thumb of participants allowing them to interact with a virtual keyboard displayed on the TV. The results of their study indicate that this approach allows for faster text entry and easier text editing, however, the effectiveness of their input method is dependent on the design and ergonomics of the remote, as well as the user's familiarity with touch-based interfaces.

A more recent study by Yang et al. [2] implemented a touch-based gesture keyboard designed to run on a smartphone. Their solution tracked the relative position of users' fingers to detect words and manipulate a keyboard displayed on the TV screen remotely. This tested the use of a *take-off* touch input approach, as opposed to the traditional *land-on* approach taken by most soft keyboards. Their gesture keyboard proved to be nearly 25% faster than a traditional touch keyboard.

2.2.3 Smartphone-TV Interaction Practices. When comparing different text input methods, touch-enabled devices consistently deliver faster input times than traditional remote controls. This is primarily due to the touch-based interactions which users are already familiar with, and which reduce the amount of movement required to manipulate the User Interface (UI). In a study by Bobeth et al. [14], both young and old age groups consistently performed faster using touch-based interactions than using either remotes or gestural interactions. Additionally, 63-72% of the participants preferred touch-based interactions to control TV's.

Smartphone-TV interactions are already becoming commonplace, and provide promising alternatives to traditional remote controls. In another survey by Deborah Torres [15], 86% of participants indicated they use their smartphone to interact with their TV at least once a month, and 38% of participants indicated they interact with their TV using their smartphone more than 3 times per week. Ease of use, speed, and convenience were the top reasons given by participants for using a smartphone instead of a remote control. The main activities performed using a smartphone were browsing TV content and searching for show titles. Participants indicated that searching on a smartphone's screen is simpler than with a remote on a TV's UI and that typing is faster and easier to perform.

3 METHOD

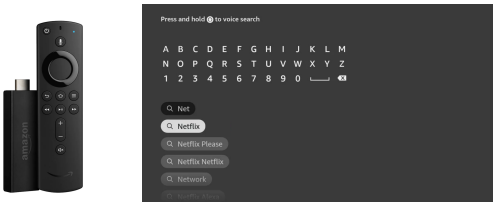

Figure 2: Amazon Fire TV Stick (left) and Amazon's built-in search interface with auto-suggestions (right).

3.1 Interaction Design

On the analysis of the conventional use of arrow keys on a remote to navigate an on-screen keyboard, input speed is limited by primarily two factors: navigational speed and character recognition. Navigation speed for these on-screen keyboards is limited due to the use of arrow keys, which require multiple concurrent key presses to navigate around the keyboard and select individual characters. Most smart TVs also use non-conventional alphabetic keyboard

layouts such as that in Figure 2 which increases the time users spend searching for and locating characters on the screen. More conventional keyboard layouts, such as the QWERTY keyboard, tend to be more familiar to users since they are commonly used for text input across most applications and devices.

Based primarily on these 2 observations, we present a novel touch-interaction-based keyboard interface as shown in Figures 3, 4, and 5. This aims to solve both limitations defined by using a more conventional QWERTY keyboard layout familiar to most users while also employing drag-based touch input for quicker keyboard navigation and manipulation. The drag-based input follows a continuous input modal allowing for quick and precise key selection guided by an on-screen cursor. This improves upon the discrete input modal utilized by TV remote buttons. Furthermore, our key selection implementation follows a *take-off* design approach based on the findings of Potter et al. [12] to allow for greater accuracy and fewer errors during input.

3.2 Visual Design

For our system's visual interface, we have incorporated several key elements, including a text-entry field, an on-screen keyboard, auto-complete suggestions, and cursors. The text-entry field is positioned at the top of the interface with the on-screen keyboard is immediately below it. Auto-complete suggestions are seamlessly integrated within the keyboard, appearing just above the QWERTY keys. The cursors, which correspond to users' touchpad input, are superimposed on the keyboard, enabling the selection of character keys and auto-complete suggestions.

3.2.1 Color Scheme. In our design, we have opted for a dark gray background, maintaining a consistent theme. To ensure optimal visibility, the text and keyboard symbols are presented in white, creating a high-contrast display against the background. In single-touch mode, the cursor adopts a teal color, while in dual-touch mode, we have assigned teal to the left cursor and red to the right cursor. This color coding provides visual feedback, as the corresponding cursor color is employed to highlight character keys or auto-complete suggestions when they are hovered over.

3.3 Hardware

Our system implementation uses a React Native application running on an Android or iOS smartphone, which connects to a Windows or macOS computer hosting a React web interface. For the scope of our experiment, mobile phones were employed as touchpad alternatives due to their convenient accessibility. Smartphones effectively emulate the mobility of a remote control, allowing users to position themselves as desired. Furthermore, mobile phone operating systems offer the experimental benefit of ease of implementation since they are programmable and support React Native, making them easier to debug compared to embedded software on a dedicated device. We connected the devices over a local network offering low latency connection.

3.3.1 Practical Implementation. For practical implementation, we recommend dedicated touchpads designed for this purpose. A dedicated device could provide advantages such as weight reduction and opportunities for adjustments in length and width.

3.4 Single-Point and Dual-Point Input

Our system supports two different input modes: single-point input and dual-point input. With single-point input, the user may control only one cursor to select keys. While with dual-point input, the user may simultaneously control two cursors thus allowing parallel key selection. As they each have their benefits and drawbacks, we aimed to explore both to understand their impact on users' experience and text-entry efficiency. We hypothesize that our dual-point touch-based input will be faster and provide a better user experience than both the single-point and traditional TV remote input methods, although it will result in a higher error rate when inputting text on a smart TV.

Figure 3: Single-Point Input Display including the phone display in portrait orientation (left) and the web interface (right).

3.4.1 Single-Point Input. For the single-point input mode, a maximum of one touch point can be used to manipulate a single teal cursor displayed on the screen as seen in Figure 3. With our systems implementation, the phone should be held in portrait mode, which is optimized for one-hand usage. Single-point input is similar to hunt and peck typing as each key is entered sequentially with only one thumb.

Advantages of Single-Point Input. With single-point text entry, a benefit is that only one hand is occupied while the other can perform another task. Compared to dual-point input, we anticipate single-point input will offer an easier learning curve as the system is more comparable to a laptop track-pad and the user only needs to focus on one thumb. This potential ease of use may result in a lower error rate, particularly for novices.

Disadvantages of Single-Point Input. One major drawback compared to dual-point input is that one finger is responsible for selecting all of the keys which may lead to fatigue. Additionally, the lone finger must be capable of reaching all of the keys. As a result, the finger will have to travel longer distances to reach the entirety of the keyboard which may affect the required touch-pad size.

Figure 4: Dual-Point Input Display including the split-screen phone display in landscape orientation (left) and the web interface (right).

3.4.2 Dual-Point Input. By introducing dual-point input, we aim to explore the potential of two-hand text entry for TVs and capitalize on the familiarity of using both hands on a traditional keyboard while typing to improve input speed through parallelism. For the dual-point input mode, two touchpoints can be used to manipulate one teal cursor and one red cursor, corresponding to the left and right touchpads respectively. These cursors are used to control text entry on the on-screen keyboard as seen in Figure 4. With our systems implementation, the phone should be held in the landscape orientation which has a similar length and width ratio to a physical keyboard.

Advantages of Dual-Point Input. With dual-point input, a benefit is that both thumbs are fully utilized for text entry. The left thumb is responsible for the keys located on the left half of the keyboard. Conversely, the right thumb is responsible for the keys located on the right half of the keyboard. This role division allows simultaneous input leading to theoretically faster text entry with the additional benefit of less distance traveled per finger as each thumb is responsible for the keys closest to it.

Disadvantages of Dual-Point Input. The main trade-off compared to single-point input is that there is possibly a higher learning curve as the user will be responsible for accurately controlling both thumbs at the same time. We hypothesize that this can potentially lead to an increase in errors. Moreover, as both hands will be used to increase the range of motion, a larger touchpad should be used to accommodate.

3.5 Relative and Absolute Positioning

Our system offers two distinct positioning modes: relative positioning and absolute positioning. In relative positioning, users employ multiple drag gestures to position the cursor before making a key selection. Conversely, with absolute positioning, users are restricted to a single drag gesture, and key selection occurs upon release. Each mode presents unique advantages and drawbacks. Due to the novel nature of absolute positioning, however, we will focus on testing the absolute positioning system to better understand its impact on user experience and text-entry efficiency.

Figure 5: Relative Positioning uses multiple drag gestures to move the cursor and a single tap gesture to select a character (left). Absolute Positioning uses a single drag gesture to set the position of the cursor on the screen. The character is selected when the finger is lifted from the screen (right).

3.5.1 Relative Positioning. Many contemporary devices equipped with touchpads, such as laptops, adopt relative positioning. This mode enables users to execute a series of drag gestures in succession to maneuver the corresponding cursor to their desired key on the on-screen keyboard. Once the correct key is highlighted, users may perform a tap action to confirm their selection, mirroring the behavior of mouse cursors on laptop touchpads. This interaction is modeled in Figure 5. The simplicity of these actions are designed to help increase accessibility for users based on the findings of Sun et al. [13]. Depending on the choice of single-point or dual-point input, the cursor may initially be positioned to the left, center, or right side of the screen. During a drag gesture, changes in positioning are measured and scaled using a multiplier for both the x and y axes, and the result is applied as an offset to the current cursor position to determine the next cursor position.

3.5.2 Absolute Positioning. Touchscreen devices, such as smartphones, tablets, and touch-enabled laptops, make use of absolute positioning. In this mode, users execute single drag gestures to reach their desired key on the on-screen keyboard, with selection occurring upon release, a behavior referred to as "take-off" [12]. This interaction is modeled in Figure 5. This same approach is commonly used in gestural keyboards such as one designed by Yang et al. [2], although in our implementation, it is used for selecting individual characters rather than strings of characters or words.

Advantages of Absolute Positioning. The primary advantage of absolute positioning lies in its efficiency for text entry. Only one drag gesture is needed for each unique character. Similarly to relative positioning, a simple tap gesture can be employed for repeated character selections. This significant reduction in the number of gestures required significantly impacts the time needed for text entry. We anticipate that experienced users may find this system to be faster compared to relative positioning.

Disadvantages of Absolute Positioning. Conversely, there are several drawbacks associated with absolute positioning. Since each drag gesture corresponds to a selection, the user is obligated to confirm a key selection every time they touch the screen, even if it is unintentional. Combined with the increased sensitivity required to navigate the full keyboard, novices may encounter higher error rates as this system deviates from convention.

Figure 6: Keyboard with auto-suggestions enabled. Suggestions dynamically update in real-time as characters are selected from the keyboard.

3.6 Auto-Complete Suggestions

Our various input modes are augmented by auto-completion suggestions integrated directly into the top of the on-screen keyboard

as can be seen in Figure 6. This feature allows partial entry on a movie title to display selectable results, thus resulting in faster text entry. We hypothesize that enabling real-time auto-complete suggestions will further improve the input speed, accuracy, and user experience across all three input modes compared to a scenario where auto-complete suggestions are disabled.

3.6.1 Auto-Complete Implementation. Our implementation of auto-complete uses an IMDb data set from which we extracted movie titles. The search suggestions return an alphabetically sorted list of titles prefixed by what the user has currently typed. We then take that list and display it above the QWERTY keys. The number of suggestions is dynamically adjusted in relation to the suggested title lengths and the length of the keyboard.

3.6.2 Benefits and Drawbacks. When enabled, auto-completion suggestions allow the user to type part of a title out and then immediately select it if it appears as a suggestion. This may prove effective for lengthy titles that require numerous key presses. However, one drawback is that the user may become distracted as they have to divert their attention to scanning through the search suggestions rather than typing the title out. Moreover, this feature consumes additional space on the user interface, potentially obscuring other content, such as thumbnails, that could be displayed instead.

4 USER STUDY

We conducted a user study to evaluate how our novel system could allow users to type faster with fewer errors. For this study, we asked participants to type and search for a set of different movie titles and measured their mean time to goal and character count. We then computed values to answer our research question: What are the effects of touch-based input methods and auto-complete suggestions on typing speed, accuracy, and usability when searching for content on a TV. Using the mean time to goal, inputted character count, and title lengths, we subsequently calculated their typing speed and accuracy across the various input methods. Furthermore, at the end of the study, we asked participants to complete a SUS survey which we used to determine usability. We then compared our proposed input methods with a standard TV remote input method to evaluate our system's effectiveness compared to traditional means of text input. Due to time limitations, we decided to evaluate our single-point absolute-positioning and dual-point absolute-positioning systems and did not have the users test or evaluate either of the relative-positioning systems. As this study serves as an initial exploration into this new text input methodology, we focus on evaluating the ability for increased efficiency and user experience and do not consider factors around implementation cost and feasibility.

Participants. The user study was conducted with a diverse group of fifteen participants, ages 18-26 including 8 males, 6 females, and 1 participant who chose not to disclose their gender. Of the participants, 10 were undergraduate students and 5 were graduate students. After a briefing of the experiment, all participants provided written consent to take place in the study. Additionally, before the study, we collected the frequency with which participants interacted with various input devices relevant to our system. The results are shown in Figure 7. This included the frequency in which they use the following related technologies: TV remotes, mouse trackpads, touch-screen devices, QWERTY keyboards, and dual-joystick game controllers. Notably, some participants reported daily use of touch-screen devices and QWERTY keyboards, while others seldom used TV remotes or game controllers, with action frequencies ranging from "never" to "multiple times a day."

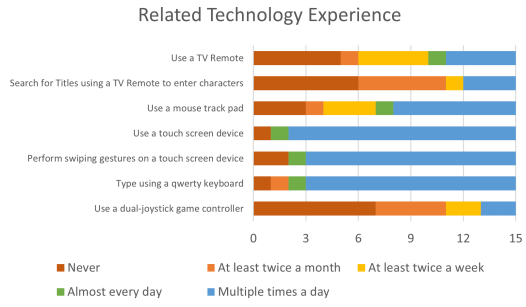

Figure 7: The results of the pre-experiment survey, which analyzed the frequency participants interacted with various relevant technologies similar to the technology used throughout the experiment.

Setup. Participants used a 6.1-inch Samsung Galaxy S22 mobile phone running Android 13 with One UI 5.1. With this device, they used our prototype app implemented on React Native to control a virtual keyboard displayed on a separate 42-inch TV. We connected this television to a laptop via HDMI, running our prototype interface built on Node.js and React. For our control variables, a separate Fire TV stick with an accompanying remote was connected to the same 42-inch TV. Interactions using the Fire TV stick were recorded using a separate web camera which was set up to view the TV, and measures were taken from reviewing the footage. Additionally, to simulate disabling text suggestions using the Fire TV, a cardboard box was used to cover the bottom of the screen where the text suggestions were displayed, and participants were informed not to use any text suggestions.

Tasks. The participants were instructed to input predetermined movie titles and initiate the search process. The selected movie titles remained consistent across all participants and included the following: "Relic," "The Silence of the Lambs," "Forrest Gump," "A League Of Their Own," and "Indiana Jones and the Last Crusade." The titles were intentionally selected to (1) cover a wide range of characters and keystroke patterns, (2) be familiar to individuals and easy to spell, and (3) appear at the same time and in the same position on both our systems and the Fire TV search system when text suggestions were enabled. This criteria is designed to aid in collecting a diverse and bias-free data set across all test cases.

Baseline System. In previous works, touch-input as a means of text entry has been studied in various novel systems [1, 2], but these methods are all experimental and do not have any widespread adoptions. As such, we have opted to use a standard TV remote method for text entry by using a Fire TV stick as seen in Figure 2 similar to that used in [1]. This method includes a built-in text-suggestion

system and allows participants to select keys and suggestions using the directional keys on a remote to navigate an on-screen keyboard and select keys using the select button on the remote. The goal of this baseline system is to provide a reference point for assessing the impact of alternative touch-based text-input methods and their impact on user input performance and overall experience.

Procedure. When participants arrived, they were given a brief introduction to the experiment and its objectives. They received a demonstration for each system used and were given time to familiarize themselves with the controls. During these demonstrations, all timing and tracking systems were disabled and the text suggestions were enabled, allowing participants to learn all aspects of each method used during the experiment. They were then allowed to freely explore each tool without any constraints on time.

Following the demonstration and training phase, participants completed 5 text entry tasks using a series of movie titles, and repeated this procedure for each of the 3 systems tested (TV Remote, Single-Point touch-input, Dual-point touch-input), both with and without text suggestions enabled, totaling 6 test conditions. To avoid any kind of carryover effects, we randomly assigned each participant a different ordering of test cases based on a balanced Latin square, as well as a different ordering of the titles to enter within each test case. As a result, we were able to collect 30 samples from each participant.

Participants were instructed to enter the title shown to them with the correct spelling. However, capitalization and punctuation were both optional. They were also informed that selecting a text suggestion or the search button should only be performed as the last action and that the resulting title should match the given title. For each task, a researcher showed the participant the title to enter and instructed them to begin once ready. The timer began the moment they first interacted with the input device and ended the moment they selected either the search button or one of the text suggestions. Data was also collected regarding the total number of characters entered, barring backspaces. These metrics were used to calculate the typing speed in words per minute (WPM) and typing accuracy (percentage of correct characters within the total number of characters input). In total, we collected 450 samples from all 15 participants.

Upon completion of the tasks for each input system, both with and without text suggestions, participants were administered a System Usability Scale (SUS) survey to evaluate the usability of the corresponding systems. Additionally, they were presented with three questions regarding the impact of text suggestions being enabled. A Likert scale with a rating of 1 (strongly disagree) to 5 (strongly agree) was employed to capture the perceived impact of text suggestions on ease of use, input speed, and error reduction. After finishing all test cases, participants completed a post-experiment survey where they ranked each of the three tested systems on several different categories including ease of use, input speed, error rate, learning curve, and personal preference. Finally, participants were allowed to provide additional, open-ended, qualitative feedback on their experience.

4.1 Results

4.1.1 Task Typing Speed. Throughout the study, we recorded the total time, in milliseconds, participants took to complete each task for the six system combinations tested: remote with suggestions, remote without suggestions, single cursor with suggestions, single cursor without suggestions, dual cursor with suggestions, and dual-cursor without suggestions. Additionally, we collected data on the number of keys entered per task which we used to calculate the rate of text entry in terms of words per minute (WPM) as seen in equation 1 below.

$$WPM = (CharactersEntered/5)/(Milliseconds/60000) \quad (1)$$

To determine the effects of each input method and the usage of text suggestions on users' typing speed, we conducted a two-way repeated measures ANOVA. The results indicate that, when controlling for text suggestions, there is a significant difference in WPM between the various input methods ($F(2, 28) = 4.42$, $p < 0.05$). It should be noted that Maulchly's Test for Sphericity indicated a violation ($W = 0.607$, $p < 0.05$), resulting in a sphericity correction of $p < 0.05$ for the effects of the input method. Similarly, when controlling for the input method, the WPM was significantly different between tasks where suggestions were and were not used ($F(1, 14) = 399$, $p < 0.001$). There was also a significant interaction effect between the input method used and whether text suggestions were used on the resulting WPM ($F(2, 28) = 3.975$, $p < 0.05$).

To compare differences in typing speed between each input method with suggestions disabled, we conducted a pairwise t-test post hoc. This showed no significant differences in the WPM between the remote and single cursor ($t(14) = -2.13$, $p = 0.15$). It also revealed that there was a significant difference in WPM between the dual cursor and remote ($t(14) = 5.2$, $p < 0.001$), as well as between the dual cursor and single cursor when suggestions were disabled ($t(14) = 3.72$, $p < 0.01$). The dual cursor ($M = 9.2$, $SD = 1.9$) was faster than both the remote ($M = 7.0$, $SD = 1.5$) and single cursor ($M = 8.0$, $SD = 1.7$) without suggestions.

Another pairwise t-test post hoc was performed to compare typing speeds between cases with text suggestions enabled. There was no significant differences in the WPM between the remote and single cursor ($t(14) = 1.41$, $p = 0.55$), and there was also no significant difference in WPM between the dual cursor and remote ($t(14) = 0.79$, $p = 1$). There was, however, a significant difference in WPM between the dual cursor and single cursor ($t(14) = 3.52$, $p < 0.05$) when suggestions were enabled, meaning the dual cursor ($M = 22.4$, $SD = 4.2$) was faster than the single cursor ($M = 19.2$, $SD = 4.9$). The remote also had an average of 21.3 wpm ($SD = 4.5$).

To compare the results of tasks with and without the presence of text suggestions, another paired t-test was used. The results indicate a significant difference in WPM between the remote without suggestions and the remote with suggestions ($t(15) = -16.4$, $p < 0.001$). There was also a significant difference in WPM between the single cursor without suggestions and the single cursor with suggestions ($t(15) = -10.5$, $p < 0.001$), as well as between the dual cursor without suggestions and the dual cursor with suggestions ($t(15) = -16.4$, $p < 0.001$).

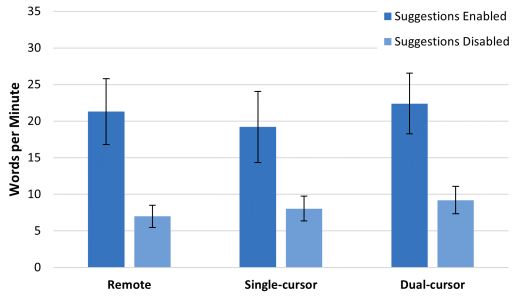

Figure 8: The average typing speed, in words per minute, participants took to complete the tasks on each input method.

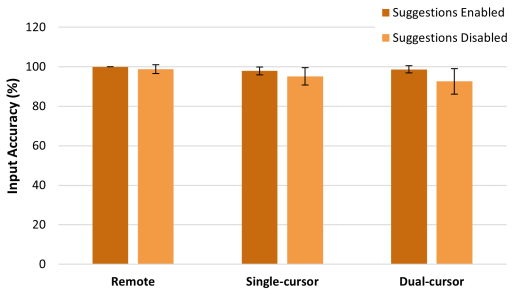

Figure 9: The average typing accuracy of participants on each input method.

4.1.2 Task Accuracy. Using the recorded number of characters entered for each task and the length of the word entered, we calculated the typing accuracy using equation 2 below.

$$Accuracy = WordLength/CharactersEntered \quad (2)$$

To determine how the input method and presence of suggestions affected text-entry accuracy, we performed a two-way repeated measures ANOVA. The examination of the input methods impact on accuracy, when controlling for text suggestions, reveals a statistically significant effect ($F(2, 28) = 12.873$, $p < 0.001$). Shifting the focus to the suggestions factor, when controlling for the input type, a substantial impact on accuracy is revealed ($F(1, 14) = 14.896$, $p < 0.05$). In the interaction's examination effect between the input method and suggestions on accuracy, a statistically significant result emerges ($F(2, 28) = 6.823$, $p < 0.01$), implying that the combined influence of these factors is not merely additive.

A pairwise t-test post hoc was performed to compare the input accuracy across the input methods used. With suggestions enabled, there was a significant difference observed between the remote and single cursor methods ($t(14) = 4.19$, $p < 0.01$), however there was no significant difference between the dual cursor and single cursor ($t(14) = 1.48$, $p = 0.48$) or between the dual cursor and remote ($t(14) = -2.71$, $p = 0.05$). The remote ($M = 100\%$, $SD = 0\%$) was more accurate than the single cursor method ($M = 97.9\%$, $SD = 1.9\%$). The dual cursor method also had an average accuracy of 98.7% ($SD = 1.9\%$) when suggestions were enabled.

Contrarily, with suggestions disabled, there was a significant difference between the dual cursor and remote ($t(14) = -4.00$, $p < 0.01$), but not between the dual cursor and single cursor ($t(14) = -2.27$, $p = 0.12$) or between the remote and single cursor ($t(14) = 2.63$, $p = 0.06$). The remote ($M = 98.8\%$, $SD = 2.3\%$) was more accurate than the dual cursor ($M = 92.6\%$, $SD = 6.5\%$). Additionally, the single cursor had an average accuracy of 95.2% ($SD = 4.4\%$) with suggestions disabled.

To compare the accuracy with and without the use of text suggestions, a pairwise t-test post hoc was performed which found a significant difference between the dual cursor with and without suggestions ($t(14) = 3.58$, $p < 0.01$) as well as between the single cursor with and without suggestions ($t(14) = 2.97$, $p = 0.01$). There was no significant difference between the remote with and without suggestions ($t(14) = 2.08$, $p = 0.06$).

4.1.3 System Usability Scales. After participants interacted and completed their tasks for all the systems assessed in the study, they were asked to provide feedback through a System Usability Scales (SUS) survey aimed at quantifying and comparing the overall usability of each system. The results revealed notable disparities in participant ratings. The single-cursor input method received an average usability score of 54.0 ($SD = 19.2$), while the dual-cursor input method achieved an average score of 50.0 ($SD = 21.3$). In contrast, participants rated the baseline remote input method significantly higher, yielding an average usability score of 69.5 ($SD = 14.6$).

To find out the statistical significance among the methods, we conducted a within participants one-way ANOVA test, which revealed a substantial difference in perceived usability ($F(2, 28) = 5.73$, $p < 0.01$). Further pair-wise comparisons utilizing paired t-tests unveiled varying degrees of usability. The dual-cursor input method exhibited significantly lower usability compared to the remote input method ($T(14) = -3.68$, $p < 0.01$). Interestingly, there was no statistically significant difference in usability between the dual-cursor and single-cursor input methods ($T(14) = -0.56$, $p = 1$). However, the remote input method demonstrated significantly higher usability than the single-cursor input method ($T(14) = 2.74$, $p < 0.05$). These findings highlight that the remote input method remains the most effective in terms of usability.

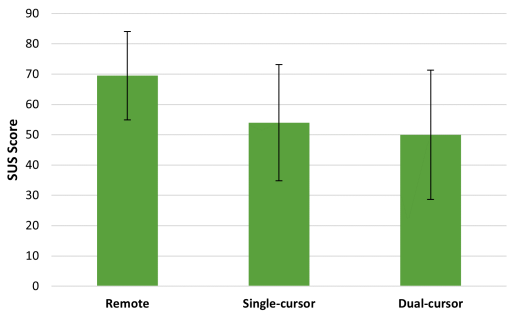

Figure 10: The System Usability Scores (SUS) calculated from participant surveys.

4.1.4 Perceived Text Suggestion Impact. In addition to a standardized SUS survey conducted for each input method, participants

also answered an additional 3 questions about the impact of text suggestions on the corresponding method. These questions asked participants to rate how they thought text suggestions impacted the overall ease of use, the typing speed, and the error rate. They rated each of these factors on a scale of 1 (very negatively) to 5 (very positively).

Participants rated the impact of text suggestions on ease of use an average of 4.0 for dual cursor input (SD = 1.3), 4.4 for TV remote input (SD = 0.8), and 4.3 for single cursor input (SD = 1.0). This indicates that text suggestions have a slightly positive impact on ease of use. For input speed, participants rated the dual cursor input method an average of 4.3 (SD = 0.6), the remote method an average of 4.0 (SD = 1.2), and the single input method an average of 4.2 (SD = 0.7), indicating that the average perceived effect text suggestions have on input speed is also slightly positive. Finally, participants rated the effect on error rate as 3.4 for dual cursor input (SD = 1.5), 4.1 for remote input (SD = 1.0), and 3.4 for single cursor input (SD = 1.4). This represents the perceived impact as slightly positive for the TV remote input method, and neither positive nor negative for both the single and dual cursor input methods.

For each of the three components measured, a separate one-way ANOVA test was performed. The results indicate that the input method had a significant effect on input speed ($F(2, 28) = 5.733$, $p < 0.01$), but not on ease of use ($F(2, 28) = 0.743$, $p = 0.485$) or error reduction ($F(2, 28) = 4.193$, $p < 0.05$). Mauchly's test indicated violations in sphericity for ease of use ($W = 0.852$, $p = 0.354$) and error reduction ($W = 0.996$, $p = 0.973$), necessitating sphericity corrections.

We used paired t-tests post hoc to to find the perceived effects of input method on input speed. We found a significant difference between the dual cursor and remote methods ($T(14) = -3.68$, $p < 0.01$). We also found significant differences between the remote and single-cursor methods ($T(14) = 2.74$, $p < 0.05$). These results emphasize the impact of input method on perceived input speed. On the contrary, we found no significant differences in ease of use and error reduction between the different input types.

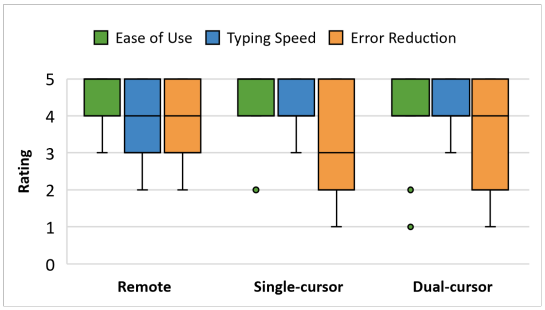

Figure 11: Participants rated how they felt text suggestions affected the ease of use, typing speed, and error rate for each input method. Answers were rated from 1 (very negative) to 5 (very positive).

4.1.5 Comparative Analysis of Systems. Figure 12 visualizes participants' responses to several subjective questions ranking each system across 5 categories including ease of use, input speed, error rate, learnability, and personal preference. Participants 8 and 14 did not complete this part of the survey, so their responses have been excluded from this comparison. For each of these categories, a ranking of 1 is better than a ranking of 3.

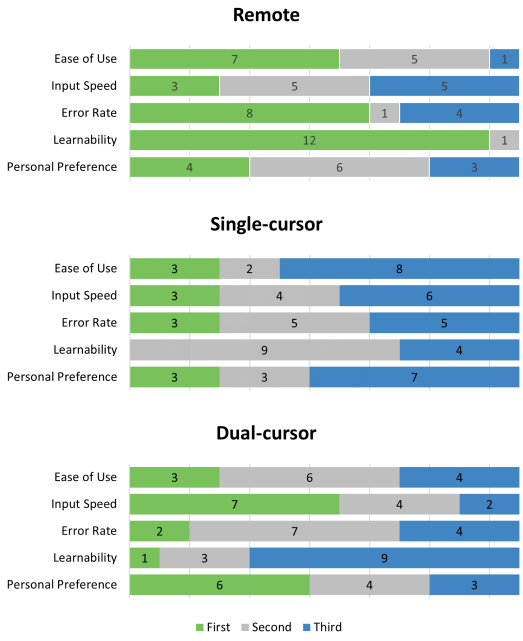

Figure 12: Results of the comparison survey conducted after our study. Each input method was ranked against each other across 5 categories to analyze perceived user metrics

5 DISCUSSION

The goal of our study was to understand the effects of the input method and suggestion mode on typing speed and accuracy. From two-way repeated measures ANOVA tests we found that both factors, input method and suggestion mode, had significant impact on both WPM and accuracy when controlling for the other. Additionally, we found that there was significant interaction between input method and suggestions implying that their combined effect is not just additive.

5.1 Comparing Input Methods

From the SUS surveys participants on average gave the remote a higher score and perceived it to be significantly more usable than both the single-cursor and dual-cursor modes from our system. Additionally, participants perceived a higher typing speed for the remote compared to the single-point and dual-point inputs. Moreover, the participants did not perceive the input method to have any significant impact on ease of use or error reduction. We then controlled for whether text suggestions were enabled or disabled to analyze the effects of each input method from the task data.

5.1.1 Remote vs. Single-Point. Regardless if suggestions were enabled or disabled we found no significant difference in WPM for

remote and single-cursor input. When suggestions were disabled we found no significant difference in accuracy. However, if suggestions were enabled the remote had significantly better accuracy than the single-cursor input.

5.1.2 Remote vs. Dual-Point. Interestingly, we found that the dual-cursor had significantly higher WPM than the remote regardless of suggestions setting. This result is contrary to what participants perceived in the SUS surveys where the remote ranked higher in WPM than the dual-cursor. Perhaps using a newly learned input method required higher focus thus contributing to a perceived increase in duration. The remote had significantly increased accuracy when suggestions were enabled. However, without suggestions, there was no difference in accuracy.

5.1.3 Single-Point vs. Dual-Point. Regardless of suggestions mode the dual-cursor input had significantly higher WPM. Additionally, regardless of suggestions setting, there was no significant difference in accuracy. These results suggest that the dual-point input may be overall better compared to the single-point input.

5.2 Comparing Suggestions Impact

According to the SUS survey results, across all three input modes, we found that text suggestions had a slightly positive impact on ease of use, input speed, and error rate. When looking at the task data, we found the typing speed was significantly faster when suggestions were enabled compared to disabled. Moreover, accuracy was significantly better with suggestions for the single-cursor input and dual-cursor input. However, we found no significant difference with the remote for both modes, as the accuracy was consistently high compared to the other input methods.

6 CONCLUSION

We proposed using touch-based controllers to interact with a virtual on-screen keyboard. By incorporating simple drag-based touch interactions on a traditional keyboard layout, our method aims to strike a balance between learnability and efficiency. Fluent single-motion key selection reduces the learning curve and is a viable substitute for existing remote-based input techniques.

We conducted a user study to evaluate our proposed solution, which employed a range of input methods, including a TV remote, single cursor, and dual cursor. We found that there was a significant difference among input methods for typing speed, with the dual-cursor method consistently outperforming the remote and single-cursor methods, especially when suggestions were disabled. Additionally, the presence of text suggestions significantly improved typing speed. We also discovered that typing accuracy was indifferent between the remote method and dual cursor method when suggestions were enabled but performed better than the dual cursor and single cursor methods when suggestions were disabled. Participants rated the remote with a higher system usability than both the single and dual cursor methods. Participants also indicated that text suggestions had a slightly positive influence on ease of use and input speed across all methods.

The comparative analysis of systems further revealed distinctions in user preferences, emphasizing the importance of considering factors such as ease of use, input speed, error rate, learnability, and personal preference when designing smart TV text entry solutions. In conclusion, no input method was found to be significantly better. Our work contributes novel approaches to this field and presents valuable findings that can inform the development of more efficient and user-friendly text input methods for smart TV interfaces.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their participation and constructive feedback during the user study.

REFERENCES

[1] Young Mi Choi and Jingtian Li. Usability evaluation of a new text input method for smart tvs. *J. Usability Studies*, 11(3):110–123, may 2016.

[2] Zhican Yang, Chun Yu, Xin Yi, and Yuanchun Shi. Investigating gesture typing for indirect touch. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 3(3), sep 2019.

[3] Zhenyu Gu, Xinya Xu, Chen Chu, and Yuchen Zhang. To write not select, a new text entry method using joystick. In Masaaki Kurosu, editor, *Human-Computer Interaction: Interaction Technologies*, pages 35–43, Cham, 2015. Springer International Publishing.

[4] Kentaro Go, Hayato Konishi, and Yoshisuke Matsuura. Itone: A japanese text input method for a dual joystick game controller. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, page 3141–3146, New York, NY, USA, 2008. Association for Computing Machinery.

[5] Andrew D. Wilson and Maneesh Agrawala. Text entry using a dual joystick game controller. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, page 475–478, New York, NY, USA, 2006. Association for Computing Machinery.

[6] Lode Hoste and Beat Signer. Speeg2: A speech- and gesture-based interface for efficient controller-free text input. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*, ICMI '13, page 213–220, New York, NY, USA, 2013. Association for Computing Machinery.

[7] Kent Wittenburg, Tom Lanning, Derek Schwenke, Hal Shubin, and Anthony Vetro. The prospects for unrestricted speech input for tv content search. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '06, page 352–359, New York, NY, USA, 2006. Association for Computing Machinery.

[8] Yiqin Lu, Chun Yu, Xin Yi, Yuanchun Shi, and Shengdong Zhao. Blindtype: Eyes-free text entry on handheld touchpad by leveraging thumb's muscle memory. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 1(2), jun 2017.

[9] Ülkü Meteriz Yýldýran, Necip Fazýl Yýldýran, and David Mohaisen. Acoustictype: Smartwatch-enabled cross-device text entry method using keyboard acoustics. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI EA '22, New York, NY, USA, 2022. Association for Computing Machinery.

[10] Steven J. Castellucci and I. Scott MacKenzie. Unigest: Text entry using three degrees of motion. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, page 3549–3554, New York, NY, USA, 2008. Association for Computing Machinery.

[11] David Goldberg and Cate Richardson. Touch-typing with a stylus. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, pages 80–87, 1993.

[12] R. L. Potter, L. J. Weldon, and B. Shneiderman. Improving the accuracy of touch screens: An experimental evaluation of three strategies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '88, page 27–32, New York, NY, USA, 1988. Association for Computing Machinery.

[13] Jiaxu Sun, Yongchao Li, Linzhang Wang, Xuandong Li, Xiaoxiao Ma, Jing Xu, and Guanling Chen. Controlling smart tvs using touch gestures on mobile devices. In *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)*, pages 1222–1229, Beijing, China, 2015. Institute of Electrical and Electronic Engineers.

[14] Jan Bobeth, Johann Schrammel, Stephanie Deutsch, Michael Klein, Mario Drobics, Christina Hochleitner, and Manfred Tscheligi. Tablet, gestures, remote control? influence of age on performance and user experience with itv applications. In *Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video*, TVX '14, page 139–146, New York, NY, USA, 2014. Association for Computing Machinery.

[15] Deborah Z. Torres. User practices for smartphone control of tv. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, MobileHCI '18, page 416–424, New York, NY, USA, 2018. Association for Computing Machinery.