Towards Personalized Head-Tracking Pointing

Muratcan Cicek
Roberto Manduchi
mcicek@ucsc.edu
manduchi@soe.ucsc.edu
University of California, Santa Cruz
Santa Cruz, California, USA

ABSTRACT

Head-based pointing is an effective interface for those with limited hand control, though it may involve a learning curve due to discrepancies between desired pointer movement and actual system response to head motion. We theorize that individuals have unique head movement patterns for similar tasks, necessitating tailored mappings from head to pointer motion. This was explored by analyzing video data of participants tracking a moving target on-screen using head movements. The study found that using a select set of facial landmarks outperforms other methods, like focusing on the nose-tip or rotation angles, in aligning head and pointer movements. Despite this, there is still notable bias inherent in the simple affine mapping model used. Significant variations were observed in participants' head movements when responding to similar target paths, with diagonal movements showing a higher error rate. These insights could be crucial in creating new, personalized head-tracking interfaces, enhancing ease and efficiency.

CCS CONCEPTS

• **Human-centered computing** → **Pointing**; **Empirical studies in accessibility**; *Accessibility technologies*; *User studies*.

KEYWORDS

Pointing, Head-based Interaction, User Study, Personalization

ACM Reference Format:
Muratcan Cicek and Roberto Manduchi. 2024. Towards Personalized Head-Tracking Pointing. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '24), May 11–16, 2024, Honolulu, HI, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3613905.3650996

1 INTRODUCTION

Standard computer interfaces were designed for people with good control of their hands (to use keyboard, mouse, track-pad, or touchscreen). A number of access technologies have been made available for those who cannot use their hand.

In many cases, touchscreen control is possible even with reduced manual dexterity [32], employing customized "touch models" that can accommodate individual abilities (e.g., touching the screen with the back or side of one's hand or with one's knuckles [22, 23], or even using one's nose [25]). For desktop or laptop computer access, which is the object of our work, several adaptations are available, such as larger/smaller keyboards, rollerball mice, and joysticks. However, these adaptations cannot be used by those with paralysis or poor control of their hands, including people with upper limb motor impairment. We should mention that another input modality, based on eye gaze, can also be used by people who cannot operate a mouse or keyboard. Eye gaze pointing, though, is generally less accepted than head pointing [5], and it is often reserved for those who cannot control their head (e.g., in the terminal stages of ALS [3]).

Individuals who cannot operate a keyboard or mouse may use physical tools such as mouthsticks or head-mounted styluses. These devices are durable and simple to use, but some may feel awkward employing them in social settings. The need for a physical device is removed when using a different interface technology: head-tracking pointing, which allows users to move the pointer on the screen using their head. Head movements can be recorded by a user-facing camera (normally embedded in screens and smartphones). Images from the camera are analyzed in real-time by a computer vision algorithm to extract relevant data features that are used to control the pointer. Existing open-source software packages that implement pointer control by head tracking include Camera Mouse [6] and Enable Viacam. Head pointing control is also available as a native accessibility feature under MacOS.

Head-tracking pointing is a powerful access modality, enabling people with poor hand control to use all the functionalities of a regular computer. Unfortunately, using current technology, typical interaction tasks are substantially slower and less accurate with head pointing than with a hand-controlled mouse, as multiple studies have shown (e.g., [17, 20, 27]). The goal driving our work is to develop a new system for head pointing control that has higher accuracy and enables faster completion of pointing tasks compared with existing systems. The main premise of our approach is that different individuals may prefer to adopt different head motion patterns to accomplish the same goal. This is particularly true for the potential users of a head pointing system, who may experience limited active neck range of motion (ROM [24]) or poor control of head movement.

Existing head pointer mechanisms pre-define a control algorithm mapping head motion to pointer motion, to which the user needs to adapt. Even though these applications normally afford a few controllable settings (gain, acceleration, smoothness), the control algorithm may not reflect the "natural" motion patterns that users would spontaneously adopt. This may result in a long learning curve [17] and relatively poor pointing performance [17, 20, 27].

CHI EA '24, May 11–16, 2024, Honolulu, HI, USA

ACM ISBN 979-8-4007-0331-7/24/05
https://doi.org/10.1145/3613905.3650996

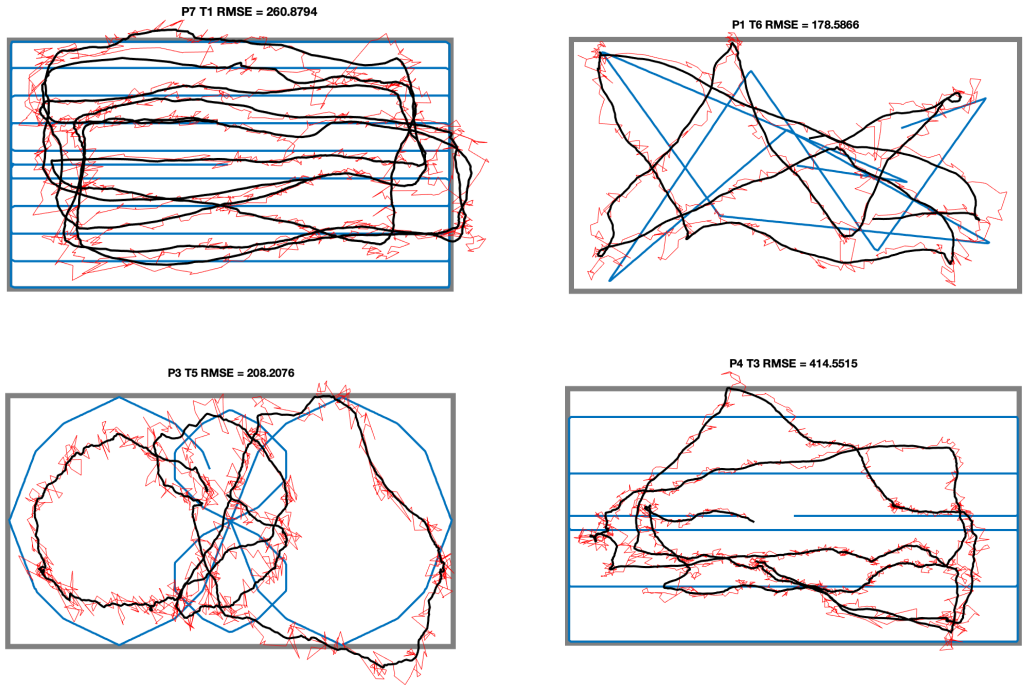

Figure 1: Examples of mapped trajectories (red), shown along with their smoothed version (black) and target trajectory (blue), for the 6_landmarks_xyz feature. Each plot displays the participant ID (e.g., P7), trajectory ID (e.g., T1), and resulting RMSE in screen pixel units. The gray frame represents the screen viewport.

Rather than stipulate a control algorithm and then tune some algorithm parameters to try to match the user's preferences, we propose a new solution that turns the problem on its head: measure and characterize each user's natural head motion patterns, then build a control algorithm around those.

Prior work by Çiçek and Manduchi[8] involved creating "trajectory videos" showing a target moving on a screen, which participants watched while mimicking the motion with their heads as if controlling the target. The study aimed to capture natural head movements without the participants directly controlling the cursor. Participants were informed about the target's future trajectory but received no other feedback. Their head movements, recorded by a camera, were proposed as indicative of natural, active head motion patterns that could be used in designing a pointer control system where the cursor moves as the user intends.

This paper presents an analysis of the data collected by Çiçek and Manduchi. We considered several types of features (different facial landmarks and geometric head pose), and for each type of feature, we computed a simple affine transformation (via least squares regression) mapping the chosen features to pointer locations on the screen. We then compared the "mapped" pointer trajectories with the trajectories of the marker the participants were following with their heads. Ideally, the mapping would reflect the user's intent, and the two trajectories would coincide. In practice, we observed large discrepancies. This should not come as a surprise: the participants did not receive any feedback about whether their head movement would map to the desired movement of the pointer. In a real application, users rely on visual feedback from the mapped pointer motion on the screen to control their own head motion.

Still, the discrepancy between mapped and desired trajectories in this "feed-forward" system, which operates by predicting and acting without receiving real-time feedback, may reveal the extent to which the considered mapping enables "natural" head motions to accomplish desired pointer movements.

The work presented here is structured into three main tasks:

- ***Task 1: Feature Selection.*** We measured the root square mean error (RSME) of trajectory discrepancy for two families of features: facial landmarks and geometric pose (3-D head rotation along with 3-D head location).
- ***Task 2: Trajectory Bias.*** We computed the spatial distribution (over 25 regions on the screen) of bias and standard deviation of discrepancy (difference between mapped and target location). Bias reveals consistent errors that could potentially be mitigated by an appropriate mapping design. Standard deviation is an inverse measure of *consistency*: whether or not a participant moved his/her head in the same way when the target trajectory was going through the same region of the screen.
- ***Task 3: Orientation.*** We measured the angular discrepancy between the velocity of the mapped and target trajectories. This analysis is especially insightful for head-pointing systems based on velocity mapping. Such mapping also enables simple resting position reset strategies and is detailed in Section 2. In addition, to account for angular discrepancies that

can result from activation of different muscle groups when moving one's head in different directions, we computed the distribution of angular bias and standard deviation over the 8 octants of the plane.

2 RELATED WORK

Prototypes of head-based pointing controllers were first created in the 1970s using ultrasound transmitters [26].

Later models (e.g., Orin Instrument's Head Mouse [16]) used a small reflective patch attached to the user's glasses or face, reflecting light from an infrared illuminator. A camera attached to the screen recorded the location of the reflective patch, which was used to control the pointer. The need for a reflective patch can be obviated by computer vision algorithms that detect specific features (e.g., the nose tip [7]) in the user's face and use the location of these features in the image plane to control the pointer [6, 14, 28]. Individual facial features, however, only provide a partial description of one's head motion. Since the head is a rigid object, its pose can be completely described by a 3-D location vector and a 3-D orientation representation.

The head pose can be determined from images using computer vision [13, 29, 33]. The advent of 3-D cameras in smartphones and tablets made head pose estimation easier and more robust [9, 10, 15].

A critical component of a head-pointing system is the mechanism used to map head motion to the pointer location on the screen (the controller.) The simplest controller design linearly maps the location of a face feature in the image plane, or the angles of head rotation, to the pointer location. The proportionality constants (gain or sensitivity) can be calibrated by means of simple procedures, e.g., by asking the user to move their head left/right and up/down, then determining appropriate gains such that the associated pointer movement spans the whole screen. More sophisticated approaches for regressing appropriate gains were studied by Lin et al. [19] and by Schaab et al. [30]. Rather than directly mapping facial features to pointer location, one can employ a derivative controller, mapping velocities of the image features to velocities of the screen pointer [7, 28]. Velocity mapping integrates the mapped velocity to obtain the pointer position at each time, enabling seamless resetting of the resting or neutral position. If, for example, the user repositioned themselves on the chair, their resting position may move slightly to one side. Resting position reset can be implemented via screen edge clipping [7]: the user moves the pointer to an edge of the screen, then moves their head as if to push it further, with this additional head motion being discarded by the controller. A non-linear function can be applied on the velocity, for example, to impose a minimum motion threshold (thus minimizing the effect of noise [7]) or to enhance pointer velocity for larger head motion (velocity-dependent gain [20]). Other proposed mechanisms include a "joystick" mode [12, 20].

Besides pointer control, other input tasks (e.g., selection, generally accomplished with a mouse click or trackpad tap) may be challenging without hand control. Several selection mechanisms are available for head pointer users. Perhaps the simplest one is selection by dwell: selection is triggered by the pointer remaining within a certain area (e.g., a checkbox) for a certain amount of time. Other options include using an external switch (e.g., implemented with sip-puff) or using facial expressions [10, 35]. For interaction with a touchscreen device, which can accept different types of gestures, multiple facial expressions can be considered [34].

3 METHOD

3.1 Video Data Set

In this study, we used the data set described in [8]. This data set contains videos of 9 participants (3 female, 6 male) taken by a screen camera as they moved their heads while imagining following a moving dot visible on the screen (*target trajectory*). All participants except for P9 had no mobility impairment. P9, who has cerebral palsy, has limited control of his limbs but regularly uses a head-pointing system (Enable Viacam) as an interface device.

Participants in the study were shown 17 short target trajectory videos with a small target (a white disk) moving along a predetermined trajectory against a black background. They were able to see the future path of the target (shown with dimmed brightness), so that they would know in advance where the target would move next. Participants were instructed to move their head while watching each video "as if" they were controlling the target with their head motion. No other instructions (e.g., how much to move their head, whether to rotate it vs. displace it, etc.) were given. Some of the target trajectories shown in the videos were repeated at a slower velocity. Some trajectories included "pause" points, where the target would stop for one second. Examples of trajectories are shown in Fig. 1.

Target trajectory videos were shown on a screen with a resolution of 1920 by 1080 pixels. Videos of the participants were taken at 1280 by 720 pixels resolution and at a frame rate of 25 Hz.

Feature type	RMSE (pixels)
nose_tip_xy	333.24
nose_tip_xyz	332.40
6_landmarks_xyz	296.71
Euler_angs	3324.88
Euler_angs_loc	307.66

Figure 2: The table shows the RMSE values (averaged over all participants) for the features considered.

3.2 Features

3.2.1 Facial Landmarks. We used mediaPipe [21] for face and hand landmark detection. We found it more practical during the implementation. Moreover, the MediaPipe framework detects 468 landmarks as opposed to the alternatives like OpenFace[4], which detects only 68 landmarks. MediaPipe uses weak perspective (scaled orthography) to compute the (x, y) coordinates of each landmark. In addition, it computes the relative depth (z coordinate) of each feature. The (x, y, z) coordinates are then rescaled by a common factor.

We consider the following features based on facial landmarks:

nose_tip_xy : This is formed by the (x, y) coordinates of MediaPipe landmark #4. The location of the nose tip in the image has been considered in prior systems described in the literature [7, 14] as well as in [8].

nose_tip_xyz : While the prior work cited above only used the 2-D nose tip location in the image, for this feature we considered all three (x, y, z) coordinates of MediaPipe landmark #4. Our purpose was to verify whether inclusion of depth (the z coordinated) could prove beneficial.

6_landmarks_xyz : This feature contains the (x, y, z) coordinates of 6 selected MediaPipe landmarks (#4, #61, #152, #159, #291, #386). These landmarks represent the nose tip, the left mouth corner, the chin tip, the left outer eye corner, the right mouth corner, and the right outer eye corner, respectively. We selected these landmarks as they form a minimal face shape with richer information than the nose tip. They implicitly indicate the head location and rotation which are more effective for determining the orientation and focus of the head.

3.2.2 Head Pose Landmarks. Since one's head can be approximately modeled as a rigid object, its pose (location + orientation) can be considered as a feature. We computed the full pose of the participant's head at each frame using the Perspective-n-Point (PnP) algorithm [18], which aligns 3D head model points to 2D facial landmarks detected in the image, providing accurate pose estimation with minimal computational overhead. Note that this computation requires camera calibration, and the calibration accuracy may affect the quality of pose estimation. We defined the following features:

Euler_angs : This feature contains the three Euler angles representing the head rotation with respect to a fixed reference frame.

Euler_angs_loc : In addition to the three Euler angles, we included here the location, with respect to a fixed frame, of a frame attached to the head, forming a 6-dimensional feature vector.

3.3 Affine Mapping

Given a feature vector $\mathbf{f}(\mathbf{t})$ at time t, we map it to a pixel location $\mathbf{p}(t) = (p_x(t), p_y(t))$ using an affine transformation: $\mathbf{p} = A\mathbf{f}(t) + \mathbf{b}$. In this equation, $\mathbf{b}$ is a 2-D vector, while A is a matrix whose dimensions vary from 2×4 for `nose_tip_xy` to 2×12 for `6_landmarks_xyz`. The parameters A and $\mathbf{b}$ are computed via least squares regression. Specifically, given the target trajectory $\hat{\mathbf{p}}(t)$, we minimize the average squared residual $\|A\mathbf{f}(t) + \mathbf{b} - \hat{\mathbf{p}}(t)\|^2$.

For each feature type, the parameters were computed individually for each participant (representing a sort of "personalization"). To minimize the risk of overfitting, we employed the leave-one-out policy: for each participant, when evaluating the mapping for a certain trajectory, the parameters were computed on the 16 remaining trajectories.

In addressing the jitter observed in the mapped locations, attributed to fluctuations in landmark localization, we employed an exponential smoothing technique. This method integrates the current location data with previously smoothed values to mitigate rapid changes, using the formula $\mathbf{p_S}(t) = (1 - \alpha)\mathbf{p_S}(t - 1) + \alpha\mathbf{p}(t)$, where α, the smoothing constant, was chosen as 0.1. This selection of α reflects a deliberate balance, prioritizing the stability of historical data over the volatility of new measurements, thereby ensuring a more consistent trajectory by dampening the effects of noise.

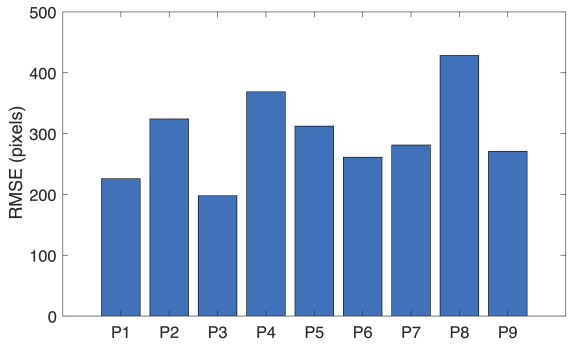

Figure 3: The bar graph at the top RMSE for all participants using feature 6_landmarks_xyz. The table at the bottom shows the RMSE values (averaged over all participants) for the features considered.

3.4 Quantitative Feature Comparison

For each feature and for each participant, we considered each target trajectory in turn, computed the affine parameters based on the remaining trajectories, and computed the residual error $\mathbf{p_S}(t) - \hat{\mathbf{p}}(t)$. We then averaged the squared norm of this error over all trajectories and took the square root of the result, obtaining one RMSE value per feature and per participant.

At this point, we tested the null hypothesis that neither participant nor feature had an effect on the RMSE values. 2-way ANOVA rejected this null hypothesis and found a significant effect of both feature and participant (in this and other tests in this article, statistical significance was set at $p \leq 0.05$). Tukey's multiple comparison test revealed a significant difference between the mean MSE obtained with `6_landmarks_xyz` and that obtained with any other feature except for `Euler_angs_loc`. Fig. 2 shows the RMSE values for each feature, averaged over all participants. On the basis of this result, we conducted the rest of our analysis using the `6_landmarks_xyz` feature, which produced the smallest average RMSE value (the distribution of RMSE across participants is shown in Fig. 3). Examples of mapped trajectories for different target trajectories and different participants are shown in Fig. 1.

3.5 Spatial Distribution of Location Discrepancies

We divided the screen area into 5×5 regions uniformly and computed the bias (average) and the total standard deviation (square root of the trace of the covariance matrix) of the error $e(t) = \mathbf{p_S}(t) - \hat{\mathbf{p}}(t)$. For each participant and for each region, the error was averaged over all t for which any target trajectory was located in that region. Note that while the bias is a 2-D vector, the total standard deviation is a scalar. For each participant and each screen region, we thus have two coordinates of bias and one value of total standard deviation. The x (y) coordinate of the screen region was found to have a significant effect on the x (y) coordinate of bias. No significant effect of region location was found on the total standard deviation. A significant effect of participants was found on the total standard deviation only. Fig. 4, left, shows the distribution of location bias and of total standard deviation across regions.

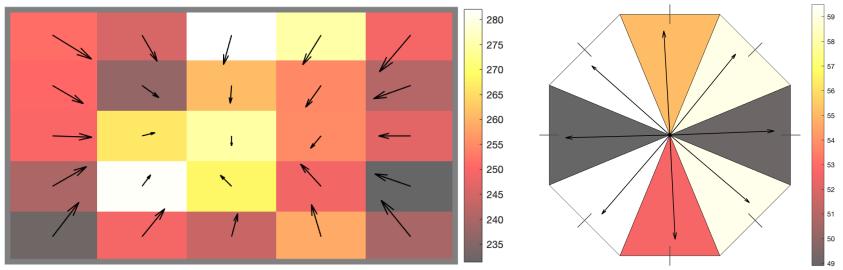

Figure 4: Top: The spatial distribution of location discrepancies. Location bias vectors are shown as arrows centered at each region. The total standard deviation at a region is indicated by the region's color. Units in the accompanying color bar are screen pixels. Bottom: The distribution of angular discrepancies across octants. The bisector of each octant is shown rotated by an amount equal to the angular bias in that octant. The color of the octant reflects the angular standard deviation in that octant. Units in the accompanying color bar are in degrees.

3.6 Angular Distribution of Velocity Discrepancies

While in the previous section we considered localization errors, here we look at the angular discrepancy between the velocity of the target and that of the mapped trajectory, which is an important consideration for controllers that are based on velocity mapping. More precisely, we considered the angular difference at each time between the tangent to the mapped and to the target trajectories distributed across octants. Specifically, for each octant $([k \cdot 45^\circ - 22.5^\circ, k \cdot 45^\circ + 22.5^\circ]$, we considered the times t in which any target trajectory $\hat{\mathbf{p}}(t)$ had tangent with a slope that was within that octant. For these intervals of time, we computed the angular difference between the slope of the tangent to the target trajectories and that to the mapped trajectories. We then computed the bias (mean) and standard deviation of these differences. This results in one value of angular bias and one value of angular standard deviation per participant and per octant. Neither participant nor octant were shown to have a significant effect on bias. Averaged across participants, the angular bias was positive for all octants (total average: 3.14°). This means that participants always move their heads in a direction slightly more counterclockwise compared to the direction of the target.

Both participant and octant had an effect on angular standard deviation. Multiple comparison analyses did not reveal a significant pairwise difference in angular standard deviation across different octants. As an additional test, we averaged together the angular standard deviation values for even-ordered and odd-ordered octants. We found a significant effect of the octant group (even- or odd-ordered) on this statistic. Fig. 4, right, shows the distribution of angular bias and standard deviation across octants.

4 DISCUSSION

4.1 Analysis of Results

Our analysis (Fig. 2) compared multiple feature types to find which of these features could be mapped (through an affine transformation) to screen points forming a trajectory that best resembles the target trajectory. A set of 6 facial landmarks (`6_landmarks_xyz`) was shown to give significantly better results (in terms of RMSE) than just the nose tip or the vector of Euler rotation angles. This type of feature can be computed robustly using modern software packages such as MediaPipe.

As shown in Fig. 4, analysis of the residual location bias (discrepancy between mapped and target trajectories) revealed a distinct spatial pattern. (Note that, although we only show results with `6_landmarks_xyz`, a similar pattern was also observed when using other features.) This clearly indicates that the simple affine transformation used to map features to screen points may not be sufficient to reproduce the intended pointer motion. Different mapping models (e.g., polynomial) could reduce or eliminate these localized biases. Importantly, the total standard deviation of the error (discrepancy) was generally very large (in excess of 200 pixels, or about 10% of the screen width). This indicates that the participants were not consistent in their head motion in response to the same target trajectory; each participant had slightly different motions. While a careful mapping function could potentially remove localized bias, this will not help with poor consistency.

Spatial inconsistency could be due to multiple reasons. For example, users may not be able to exactly replicate a certain head pose due to proprioceptive bias [1, 2, 31]. It is also conceivable that trying to reach a certain head position/orientation starting from different points may result in slightly different trajectories of head movement. If, for example, one is trying to move the pointer to the upper left corner of the screen, different neck muscle groups would be activated depending on whether the pointer is currently in the lower left area (extension), in the top right area (axial rotation), or in the center of the screen (motion around an oblique axis). Activation of different muscle groups may affect the velocity and precision of coordinated motion [27]. We conjecture that more complex sequence-to-sequence maps (e.g., recurrent neural networks) may help mitigate the lack of consistency by modeling different trajectories of head motion, thus producing a closer approximation to the intended pointer location at each time.

Our analysis of the distribution across octants of the angular error revealed that its standard deviation is larger for "diagonal" octants (Fig. 4, right). Moving the pointer along the "horizontal" or "vertical" octants requires moving one's head left/right (rotation) or up/down (extension/flexion). In each of these movements, only one group of muscles is generally activated. However, for the "diagonal" octants (diagonal head circumduction) multiple neck muscles need

to work in coordination, which may be the cause for the observed reduced consistency in these movements.

4.2 Limitations

It is important to note that the data being analyzed was collected without any feedback to the participants, who were simply moving their heads as if controlling the target with their head motion. In practice, the user of a head-pointing system relies on visual feedback to make the necessary adjustments to the pointer location. For example, if a user rotates their head to the right to move the pointer to a checkbox, and the pointer overshoots the desired location, the user would then slightly move their head to the left to compensate for it. Still, it is crucial that the controller should reflect the user's intention as much as possible to avoid the need for continuous adjustments, which can make the whole experience of computer interaction slow and frustrating.

A limitation of this data set is that the data only included "smooth" pointing tasks, with participants pretending to move a target with constant velocity along specific trajectories. This type of motion could be representative of tasks such as drawing or repositioning a window on the screen. A much more common interaction task is point-select [11], whereby the user moves the pointer from a certain location to reach a target and then selects it (normally, via a mouse click).

The video trajectories considered are poorly representative of point-select tasks, and new data collection would be necessary to study head motion in these cases.

An obvious limitation of this data set is that all participants except for one (P9) had no mobility impairments. Since the intended users of head-pointing systems are people with mobility limitations, it will be important to acquire similar data for this community of users.

5 CONCLUSIONS

We presented an analysis of the discrepancy between the intended target trajectories and mapped trajectories using a simple affine transformation of selected features from images of participants' heads. This analysis was based on an existing data set with videos of users moving their heads while following a moving target on the screen. Our analysis has shown that a set of 6 facial landmarks is superior, in terms of mapping error, to other commonly used features (nose tip, head rotation). We also reported mapping errors (in terms of bias and standard deviation) for both mapped locations and mapped velocities (angular error). In future work, we will consider more complex mapping using machine learning in an effort to ensure that the mapped trajectory faithfully reflects the user intent.

REFERENCES

[1] Bridget Armstrong, Peter McNair, and Denise Taylor. 2008. Head and neck position sense. *Sports medicine* 38 (2008), 101–117.

[2] Neil J Artz, Michael A Adams, and Patricia Dolan. 2015. Sensorimotor function of the cervical spine in healthy volunteers. *Clinical biomechanics* 30, 3 (2015), 260–268.

[3] Laura J Ball, Amy S Nordness, Susan K Fager, Katie Kersch, Brianae Mohr, Gary L Pattee, and David R Beukelman. 2010. Eye gaze access of AAC technology for people with amyotrophic lateral sclerosis. *Journal of medical speech-language pathology* 18, 3 (2010), 11.

[4] Tadas Baltrušaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. 2018. OpenFace 2.0: Facial Behavior Analysis Toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, Institute of Electrical and Electronics Engineers, Piscataway, NJ, United States.

[5] Richard Bates and Howell O Istance. 2003. Why are eye mice unpopular? A detailed comparison of head and eye controlled assistive technology pointing devices. *Universal Access in the Information Society* 2 (2003), 280–290.

[6] Margrit Betke, James Gips, and Peter Fleming. 2002. The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities. *IEEE Transactions on neural systems and Rehabilitation Engineering* 10, 1 (2002), 1–10.

[7] Muratcan Cicek, Ankit Dave, Wenxin Feng, Michael Xuelin Huang, Julia Katherine Haines, and Jeffry Nichols. 2020. Designing and evaluating head-based pointing on smartphones for people with motor impairments. In *Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility*. Association for Computing Machinery, New York, NY, United States, 1–12.

[8] Muratcan Cicek and Roberto Manduchi. 2022. Learning a Head-Tracking Pointing Interface. In *International Conference on Computers Helping People with Special Needs*. Springer, Springer International Publishing, Cham, 399–406.

[9] Muratcan Cicek, Jinrong Xie, Qiaosong Wang, and Robinson Piramuthu. 2020. Mobile Head Tracking for eCommerce and Beyond. In *IS&T International Symposium on Electronic Imaging 2020: Mobile Devices and Multimedia: Enabling Technologies, Algorithms, and Applications proceedings* (2020-01-26). Society for Imaging Science and Technology, Society for Imaging Science and Technology, San Francisco, CA, USA, 303–1–303–11.

[10] Justin Cuaresma and I Scott MacKenzie. 2017. FittsFace: Exploring navigation and selection methods for facial tracking. In *Universal Access in Human–Computer Interaction. Designing Novel Interactions: 11th International Conference, UAHCI 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9–14, 2017, Proceedings, Part II 11*. Springer, Springer International Publishing, Cham, 403–416.

[11] Sarah A Douglas, Arthur E Kirkpatrick, and I Scott MacKenzie. 1999. Testing pointing device performance and user assessment with the ISO 9241, Part 9 standard. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 215–222.

[12] D. G. Evans, S. Pettitt, and P. Blenkhorn. 1996. A head operated "joystick". In *Proceedings of the 5th International Conference on Computers Helping People with Special Needs. Part I* (Linz, Austria) *(ICCHP '96)*. R. Oldenbourg Verlag GmbH, DEU, 85–91.

[13] Yun Fu and Thomas S Huang. 2007. hMouse: Head tracking driven virtual computer mouse. In *2007 IEEE Workshop on Applications of Computer Vision (WACV'07)*. IEEE, Institute of Electrical and Electronics Engineers, Piscataway, NJ, United States, 30–30.

[14] Dmitry O Gorodnichy and Gerhard Roth. 2004. Nouse 'use your nose as a mouse'perceptual vision technology for hands-free games and interfaces. *Image and Vision Computing* 22, 12 (2004), 931–942.

[15] Sebastian Hueber, Christian Cherek, Philipp Wacker, Jan Borchers, and Simon Voelker. 2020. Headbang: Using head gestures to trigger discrete actions on mobile devices. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services*. Association for Computing Machinery, New York, NY, United States, 1–10.

[16] Origin Instruments. 2024. HeadMouse Nano. https://www.orin.com/access/headmouse/. Accessed: January 1, 2024.

[17] Richard J Jagacinski and Donald L Monk. 1985. Fitts' Law in Two dimensions with hand and head movements movements. *Journal of motor behavior* 17, 1 (1985), 77–95.

[18] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. 2009. EPnP: An Accurate O(n) Solution to the PnP Problem. *International Journal of Computer Vision* 81, 2 (2009), 155.

[19] Mei Li Lin, Robert G Radwin, and Gregg C Vanderheiden. 1992. Gain effects on performance using a head-controlled computer input device. *Ergonomics* 35, 2 (1992), 159–175.

[20] E LoPresti, DM Brienza, J Angelo, and L Gilbertson. 2000. Computer head controls: ergonomics and effect of neck movement limitations. In *Proceedings of CSUN 15th annual conference on technology and persons with disabilities, California State University, Northridge*. Association for Computing Machinery, New York, NY, USA, 147––148.

[21] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. 2019. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172* abs/1906.08172 (2019).

[22] Martez E Mott, Radu-Daniel Vatavu, Shaun K Kane, and Jacob O Wobbrock. 2016. Smart touch: Improving touch accuracy for people with motor impairments with template matching. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. Association for Computing Machinery, New York, NY, USA, 1934–1946.

[23] Martez E Mott and Jacob O Wobbrock. 2019. Cluster Touch: Improving touch accuracy on smartphones for people with motor and situational impairments. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–14.

[24] Kevin T Patton. 2018. *Anatomy & Physiology (includes A&P Online course) E-Book*. Elsevier Health Sciences, Amsterdam, Netherlands.

[25] Ondrej Polacek, Thomas Grill, and Manfred Tscheligi. 2013. NoseTapping: what else can you do with your nose?. In *Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia*. Association for Computing Machinery, New York, NY, USA, 1–9.

[26] Ronald E Prior. 1976. MOBILITY AIDS FOR THE SEVERELY HANDICAPPED A STATUS REPORT" Charles M. Scott, BSME President. *Bulletin of Prosthetics Research* 26 (1976), 192.

[27] Robert G Radwin, Gregg C Vanderheiden, and Mei-Li Lin. 1990. A method for evaluating head-controlled computer input devices using Fitts' law. *Human factors* 32, 4 (1990), 423–438.

[28] Maria Francesca Roig-Maimó, Cristina Manresa-Yee, Javier Varona, and I Scott MacKenzie. 2016. Evaluation of a mobile head-tracker interface for accessibility. In *Computers Helping People with Special Needs: 15th International Conference, ICCHP 2016, Linz, Austria, July 13-15, 2016, Proceedings, Part II 15*. Springer, Springer International Publishing, Cham, 449–456.

[29] Nataniel Ruiz, Eunji Chong, and James M Rehg. 2018. Fine-grained head pose estimation without keypoints. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. Institute of Electrical and Electronics Engineers, Piscataway, NJ, United States, 2074–2083.

[30] John A Schaab, Robert G Radwin, Gregg C Vanderheiden, and Per Krogh Hansen. 1996. A comparison of two control-display gain measures for head-controlled computer input devices. *Human Factors* 38, 3 (1996), 390–403.

[31] JL Taylor and DI McCloskey. 1988. Proprioception in the neck. *Experimental Brain Research* 70 (1988), 351–360.

[32] Shari Trewin, Cal Swart, and Donna Pettick. 2013. Physical accessibility of touchscreen smartphones. In *Proceedings of the 15th international ACM SIGACCESS conference on computers and accessibility*. Association for Computing Machinery, New York, NY, USA, 1–8.

[33] Tsun-Yi Yang, Yi-Ting Chen, Yen-Yu Lin, and Yung-Yu Chuang. 2019. Fsa-net: Learning fine-grained structure aggregation for head pose estimation from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. Institute of Electrical and Electronics Engineers, Piscataway, NJ, United States, 1087–1096.

[34] Xuan Zhao, Mingming Fan, and Teng Han. 2022. "I Don't Want People to Look At Me Differently" Designing User-Defined Above-the-Neck Gestures for People with Upper Body Motor Impairments. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3491102.3517552

[35] Rafael Zuniga and John Magee. 2017. Camera Mouse: dwell vs. computer vision-based intentional click activation. In *Universal Access in Human–Computer Interaction. Designing Novel Interactions: 11th International Conference, UAHCI 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9–14, 2017, Proceedings, Part II 11*. Springer, Springer International Publishing, Cham, 455–464.

Received 25 January 2024; revised 28 March 2024