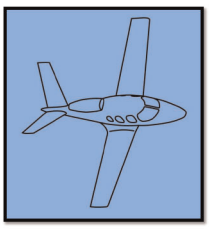
Available online at http://docs.lib.purdue.edu/jate



JATE

Journal of Aviation Technology and Engineering 10:2 (2021) 34–50

Eye-Gaze-Controlled HMDS and MFD for Military Aircraft

L.R.D. Murthy[1], Abhishek Mukhopadhyay[1], Somnath Arjun[1], Varshith Yelleti[2], Peter Thomas[3], Dilli Babu Mohan[4], and Pradipta Biswas[1]

[1]*Indian Institute of Science*
[2]*Indian Institute of Information Technology*
[3]*University of Hertfordshire*
[4]*Indian Air Force*

Abstract

Eye-gaze-controlled interfaces allow the direct manipulation of a graphical user interface by looking at it. This technology has great potential in military aviation, in particular, operating different displays in situations where pilots' hands are occupied with flying the aircraft. This paper reports studies on analyzing the accuracy of eye-gaze-controlled interfaces inside aircraft undertaking representative flying missions. We report that using eye-gaze-controlled interfaces, pilots can undertake representative pointing and selection tasks at less than two seconds on average in a transport aircraft. Further, we analyzed the accuracy of eye-gaze-tracking glasses under various G load factors and analyzed the failure modes. We observed that the accuracy of the eye-tracking glasses is less than 5° of visual angle up to +3G, although less accurate at −1G and +5G. We also found that existing eye tracker fails to track eyes under higher external illumination and needs to have a larger vertical field of view than the presently available systems. We used this analysis to develop eye-gaze trackers for multi-functional displays and head-mounted display system (HMDS). We obtained significant reduction in pointing and selection times using our proposed HMDS compared to a traditional thumb-stick-based target designation system.

Keywords: eye-gaze tracking, HMDS, aviation UI, cockpit design, MFD

1. Introduction

Eye tracking is the process of measuring either the point of gaze where one is looking or the motion of an eye relative to the head. This paper investigated the use of eye-gaze trackers in a military aviation environment as a direct controller of various types of user interfaces like head-down and head-mounted display systems.

Presently, eye-gaze-tracking devices are readily available and have already been used to directly control user interfaces. Eye-gaze-controlled displays are mainly explored for assistive technology to make computers accessible to people with severe physical impairment (Biswas & Langdon, 2013). They have also found application to facilitate interaction for touchscreen or mouse (Zhai et al., 1999). In recent times, eye-gaze-controlled interfaces have been explored for automotive user interfaces (Biswas, 2016) and De Reus et al. (2012) proposed the use of eye-gaze trackers as direct controllers of head-mounted display systems (HMDSs). Use of eye-gaze tracking to analyze pilots' interaction with cockpit displays dates back

https://doi.org/10.7771/2159-6670.1235

to the 1950s (Fitts et al., 1950; Thomas et al., 2015). Eye tracking has already been used for flight safety in the following ways (Calhoun & Janson, 1991; Fitts et al., 1950; Rudi et al., 2019; Shree et al., 2018):

- Comparing scan paths and fixation durations to evaluate the progress of pilot trainees.
- Estimating pilots' skills.
- Analyzing crew's joint attention and shared situational awareness.
- Displaying a notification at the point of a pilot's gaze to ensure visual processing, performing an automatic maneuver, and so on.

Eye-gaze-controlled interfaces have great potential for military aviation as pilots find it difficult to use existing target designation systems in high-G situations, and direct voice input systems are not well explored for non-native English speakers and for languages other than English (Eurofighter, 2017). Additionally, eye-gaze trackers can also be used to automatically estimate pilots' cognitive load (Babu et al., 2019; Shree et al., 2018). However, eye-gaze-controlled interfaces need to be evaluated in actual flight conditions as earlier studies (Borah, 1995; Calhoun & Janson, 1991; De Reus et al., 2012) only used them in simulators. Adelstein et al. (2008) reported "significant degradations in both error rate and response time in a reading task at 0.5 and 0.7 g for 10-pt, and at 0.7 g for 14-pt font displays."

In the study reported in this paper, initially we investigated the effectiveness of using eye-gaze trackers for undertaking representative pointing and selection tasks in a transport aircraft during different phases of flight on a display in head-down configuration. This configuration is similar to the setup of multi-functional displays (MFDs) in actual flight cockpits. Next, we analyzed accuracy and failure modes of eye-gaze trackers under constant-G maneuvers in BAES Hawk Trainer and Jaguar Darin aircraft and analyzed accuracy and failure modes of eye-gaze tracking.

Based on our results and analysis, we present two eye trackers developed for a simulated HMDS using a wearable eye tracker and for MFDs using a screen-mounted camera. For HMDS, we developed a multimodal head and eye-gaze tracking system and integrated it with a flight simulator. Our studies showed that our system can statistically significantly reduce target locking duration compared to a traditional target designation system (TDS). For the screen-mounted camera, we used an intelligent algorithm for facial feature point detection and gaze estimation. We compared the new algorithm with existing screen-mounted gaze tracker in different lighting conditions.

2. Literature Survey

Rudi et al.'s (2019) survey with pilots suggested that the potential advantages they foresee by using gaze-controlled interfaces are the faster access to information, increased system overview, and increased situational awareness. Smyth (1997) proposed a system with a head-up display (HUD) with electromagnetic head movement tracking and interaction using eye tracker control. However, in order to interact with the HUD, i.e., a display in the line of sight of the pilot, using eye tracking, a pilot has to calibrate initially. If one chooses to interact with a HUD using a wearable eye tracker, the gaze points from the wearable eye tracker are obtained with respect to a fixed head position. This affects the performed calibration once the pilot changes the head position or orientation. This issue of interacting with a HUD has not been considered in earlier works. The present BAES Striker II (2020) and Elbit Systems DASH (2020) helmet systems provide HMDSs using opto-inertial sensors to track head movement and adapt the content on the display accordingly. These present HMDSs also enable pilots to lock on a target by head movement. In the case of HMDS interaction, the present systems require more input besides head movement when there is more than one target in a single line of sight. Presently, targets are automatically prioritized, and pilots select the target to engage by using a flip switch. Besides, there can be only so much information that the HMDS display can accommodate without cluttering the visual field. Placing the information on an extended virtual display canvas and facilitating interaction with this content via HMDS can provide a lot more information in a structured and clutter-free manner to the pilot for gaining system overview and situational awareness. In this direction, we propose to use eye gaze directly to engage a target while there are multiple targets in one line of sight or/and for interacting with the content on the HMDS screen. We proposed an algorithm to integrate both head orientation and eye-gaze information into a single datum which can be used for selecting multiple targets in a given line of sight. With the help of the proposed system, one of the disadvantages perceived in Rudi et al. (2019), "too much information," can be overcome by facilitating a virtually larger canvas and allowing pilots to customize the information they need on the specific locations of HMDS. Shree et al. (2018) reported the use of an eye-gaze-controlled interface in a flight simulator, although that system did not allow the user to have free head movement for interaction. In this paper, we propose a multimodal eye gaze and head interface, which supports natural head movement along with eye gaze to interact with the HMDS.

In the following sections, we first describe our data collection inside actual aircraft followed by design, development, and evaluation of eye-gaze-controlled systems.

3. Study on Transport Aircraft

Although we aimed to develop an eye-gaze-controlled interface for fighter aircraft, we started data collection on a

transport aircraft for initial validation and feasibility of the system. The following study involved an ISO 9241 pointing task inside a transport aircraft. In particular, we compared two different versions of eye-gaze-controlled interface—one version only moves a pointer in a graphical user interface using eye gaze (non-adaptive); the other version not only moves a pointer but also activates a target nearest to the eye gaze location (adaptive). Details of the pointer movement algorithms are described in Shree et al. (2018).

We collected data from three Indian Air Force pilots with ranks ranging from squadron leader to wing commander. We collected data using a Microsoft Surface Pro tablet running Windows 10 operating system and a Tobii PCEye Mini Eye Tracker (2018). The eye tracker has an accuracy of 0.4° of visual angle in a desktop computing environment. We used an Avro HS748 transport aircraft for data collection purpose. We used an X16-1D USB accelerometer from Gulf Coast Data Concepts for recording vibration in units of g (g = 9.81 m/s^2). We set up the tablet and eye-gaze tracker at the front seat outside the cockpit as shown in Figure 1.

The ISO 9241 pointing task required the user to move the cursor from a center button on the screen to one of the target buttons with a preset size (W) located at a preset distance (D). We mapped the user's eye-gaze movement to the cursor and we performed our experiments with the following sizes and distances for the target buttons:

- Target sizes (W, in cm): 1.9, 1.7, 1.5, 1.3, 1.1, 0.9.
- Distance of target buttons from center of screen (D, in cm): 5, 8.

We designed a repeated measure study with the following independent variables:

- *Place of study*
  - On ground
  - In air
- *Type of system*
  - Non-adaptive
  - Adaptive, with nearest-neighborhood algorithm that activates target nearest to gaze location.

We also used an accelerometer in front of the tablet computer to record vibration while flying. In total, we analyzed 956 pointing tasks with at least 150 tasks recorded for each condition. We calculated average movement time for all possible indices of difficulty, $ID = \log_2\left(\frac{D}{W} + 1\right)$, for all different conditions. The ID is a measure of task difficulty to point at a given target with size W present at distance D from the center button. Figure 2 plots the movement times with respect of IDs for all four conditions. We found correlation coefficients r = 0.64 and r = 0.63 between movement time and ID for the non-adapted versions on ground and in air, respectively. However, with the nearest-neighborhood algorithm, the correlation coefficients were less than 0.3.

We undertook a *place of study (2)* × *type of system (2)* repeated measure ANOVA on the movement times. We found

- A significant main effect of *place of study* $F(1, 10) = 14.38$, $p < 0.05$, $\eta^2 = 0.59$.
- A significant main effect of *type of system* $F(1, 10) = 34.80$, $p < 0.05$, $\eta^2 = 0.78$.
- An interaction effect of *place of study* and *type of system* $F(1, 10) = 7.78$, $p < 0.05$, $\eta^2 = 0.44$.

A set of pairwise comparisons found that there are significant differences at $p < 0.05$ in movement times between data collected on ground and in air and between adapted and non-adapted conditions on data collected in air.

In terms of qualitative feedback, all pilots preferred the adaptive version over the non-adaptive one. In particular, they noted that the non-adaptive version becomes difficult to use during take-off and landing phases compared to cruising phase. In terms of application, they noted that the system will be useful for operating the MFD and operating the HMDS for investigating and engaging beyond-visual-range targets.

We further analyzed the cursor movement trajectories for both adaptive and non-adaptive conditions. Cursor movement efficiency can be analyzed in detail using the metrics defined by MacKenzie et al. (2001) with regards to the task axis (the line between starting source and intended target). These metrics look at the variability of movements as well

Figure 1. Aircraft used in the study and placement of setup inside the aircraft.

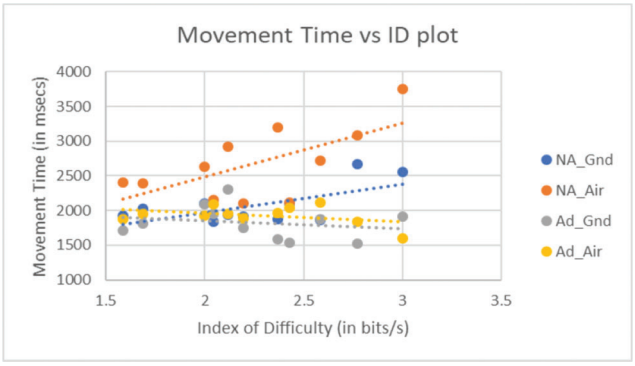


Figure 2. Movement time versus ID plot.

as the number of events relating to cursor movement along the task axis towards the intended target. These are illustrated in Figure 3.

Movement variability (MV) is defined as the standard deviation in the variation in orthogonal movement, relative to the average deviation:

$$MV = \sqrt{\frac{\sum_{i=1}^{n} (y_i' - \bar{y}')^2}{n-1}}.$$

Movement error (ME) concerns the overall mean magnitude of deviation in cursor movement from the task axis:

$$ME = \frac{\sum_{i=1}^{n} |y_i'|}{n}.$$

The movement offset (MO) is the average magnitude of deviation, $\bar{y}'$. Both the orthogonal and movement direction change metrics (ODC and MDC, respectively) indicate the number of times the participant changes the direction of cursor movement orthogonal to, and parallel to, the task axis, respectively, as the cursor is moved towards the intended target. The task axis crossing (TAC) concerns the number of times the cursor is moved across the task axis as the cursor is moved towards the target. Lastly, target re-entry (RE) counts the number of times the cursor moves back into the target area after first leaving.

Initially we compared the average values of these parameters in adaptive and non-adaptive conditions (Figure 4). We used a * marks on the graph for parameters, which were significantly different at $p<0.05$ in an unequal variance t-test.

Furthermore, we analyzed the data with two more visualization techniques for obtaining better insight about the information, which is not feasible with the plots described above. We used radial stacked bar chart and developed a novel approach called scattered radial bar plots. Using radial stacked bar chart (Abel & Sander, 2014; Stasko & Zhang, 2000), we can view and compare all dependent variables for adaptive and non-adaptive data in one visual frame against all IDs together instead of plotting each dependent variable separately against each ID. From Figure 5 it may be noted from the extent of the gray and

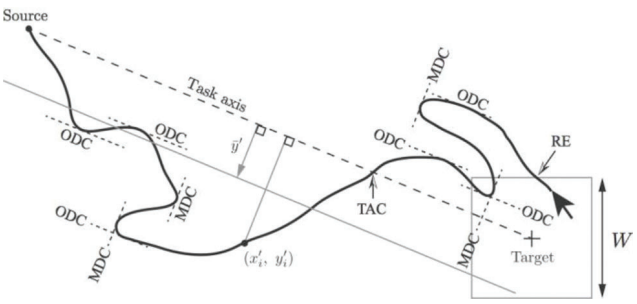


Figure 3. Illustration of cursor efficiency metrics.

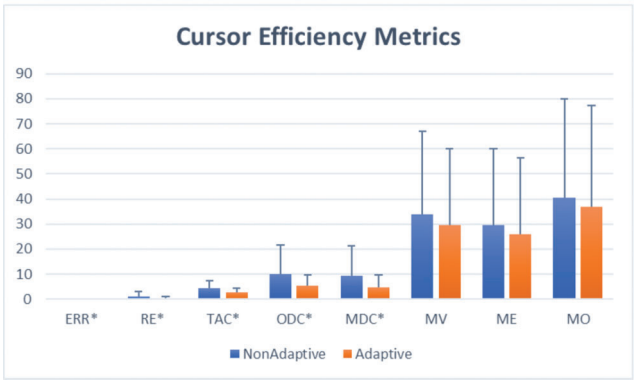


Figure 4. Comparing cursor efficiency metrics between adaptive and non-adaptive conditions. Asterisks mark parameters that were significantly different at $p < 0.05$ in an unequal variance *t*-test.

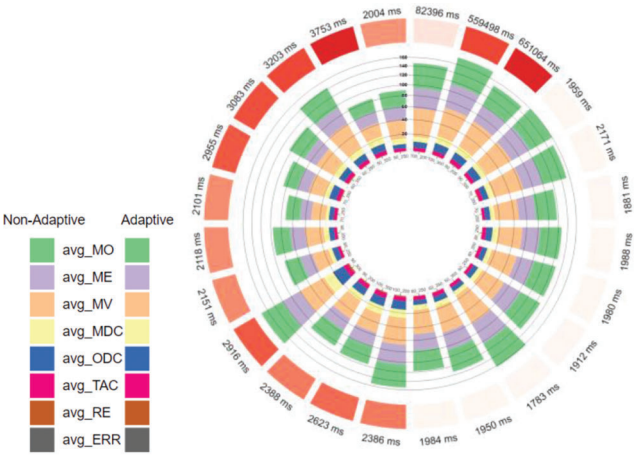


Figure 5. Radial stacked bar chart.

brown part that the average movement offset (avg_MO) and average movement error (avg_ME) for target width 100 and distance 200 are larger than the other IDs. We also observed that avg_ME for target width 100 and distance 300 is largest in adaptive data, but in non-adaptive data target width 90 and distance 300 has the largest avg_ME. In the non-adaptive part of the figure, effectiveness of all the dependent variables cumulatively for target width 90 and distance 300 is the largest amongst all values of IDs, and for the adaptive section of the visualization the overall effectiveness of all the dependent variables together for target width 50 and distance 350 and target width 100 and distance 300 are the largest despite the time taken to complete the task being different for both. From the graphs, we can note that ME and MV have different values for different IDs while RE and Error were not much affected by different values of IDs. Using scattered radial bar plots (Chernoff, 1973; Stasko & Zhang, 2000), we examined dependent variables for each ID individually. For example, in Figure 6 we can comprehend that average error (avg_ERR) for target width 90 and distance 200 is more compared to other IDs, which is evident from the green part

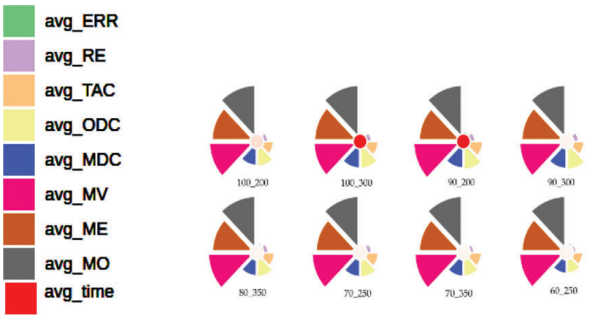


Figure 6. Scattered radial bar plots.

Figure 7. Sample vibration profile of the aircraft.

of the bar in the plots. We also noticed that average movement offset (avg_MO), average movement error (avg_ME), and average movement variability (avg_MV) which are represented by gray, brown, and pink colors, respectively, for all the IDs are almost similar irrespective of the time.

We separately analyzed the vibration profile (Figure 7) in terms of the acceleration values recorded for roll, pitch, and yaw. The roll and yaw vibration had a maximum value of 1.2G while the acceleration measured for pitch reached 1.5G.

This study shows that the nearest-neighborhood algorithm made selection of smaller targets easier as indicated by the low value of correlation between movement time and ID. This ease of selection of small targets turns out to be more useful in air under vibrating condition than on ground as indicated by the ANOVA study and pairwise comparisons. It may also be noted that using the nearest-neighborhood algorithm, participants can select target using gaze-controlled interface in less than 2 seconds on average both on ground and in air. Analysis of the cursor efficiency metrics shows that the nearest-neighborhood algorithm reduces the homing time to target as indicated by the significantly lower values of RE, TDC, ODC, and MDC but at a cost of higher error rate, although it is less than 5%.

In this study, we used two different devices for measuring movement time and acceleration and hence cannot synchronize them at the milliseconds level. We could not make separate analyses for different flying phases, and, being in a transport aircraft, we could measure performance of the gaze-controlled system up to 1.5G only. In our future studies, we are planning to collect data on a fighter aircraft attaining higher G values and synchronizing the users' performance with vibration profiles.

4. Study on Fighter Aircraft

In this study, we collected data from two fighter aircraft. The first, a BAE Systems Hawk, is a British single-engine, jet-powered, twin-seater trainer aircraft in tandem seating configuration. The second, a Jaguar aircraft, is used for close air-support and ground-attack missions. The Hawk aircraft is flown by a pilot (age 35 years) with a flying experience of 1920 hours in multirole combat aircraft and the Jaguar aircraft is flown by a pilot (35 years) with a flying experience of 2100 hours in bomber aircraft.

We recorded data from two flights using a commercial off-the-shelf (COTS) eye tracker (Tobii Pro Glasses 2) which uses infrared (IR) illumination-based eye-gaze estimation principles (Tobii Pro Glasses 2 Product Description, 2019). The duration of the first flight is 55 minutes 58 seconds (Flight 1) and the other flight's duration is 56 minutes (Flight 2). The flight profiles are provided below.

- **Flight #1** Maneuvering flight with head-mounted eye tracker on pilot in command. Take-off—climb—level flight to local flying area—constant G (3G and 5G) level turns both sides each—vertical loop—barrel roll—air-to-ground dive attack training missions—descent—ILS approach and landing.
- **Flight #2** Non-maneuvering flight with head-mounted eye tracker on pilot in command. Take-off—climb—level flight to local flying area—straight and level cruise with gentle level turns—descent—ILS approach and landing.

Flight #1 is recorded in the Jaguar aircraft while Flight #2 is recorded in the Hawk aircraft. Our eye-tracking data are recorded in constant G levels. We collected data at +5G, +3G, and −1G and compared with data collected at +1G. A demonstration video can be found in supplementary material 1.

We used the Tobii Pro Glasses (2019) for data collection, having one scene camera to record outside view and four other cameras, two for each eye to record eye gaze at 100 Hz. A proprietary software (Tobii Pro Studio) maps eye gaze on the video recorded in the scene camera and indicates the point of gaze fixation by drawing a red circle on the scene video. The recorded point of fixation is referred to as the gaze point in subsequent analysis. The eye tracker contains a front-facing scene camera which records the first-person view of the pilot. It also contains four eye cameras, two cameras per eye, to record the eye movements. The eye tracker estimates gaze points at a frequency of 100 Hz. The frame rate of the scene camera is 25.01 frames/second at 1920 × 1080 resolution and that of each eye camera is around 50 frames/second with a resolution of 240 × 240. Each gaze point is recorded with a dedicated identifier, called "gidx." We initially used Tobii Pro Lab tool to analyze the recorded gaze samples and observed that

both flight recordings contain gaze samples only for around 50% of the duration. We investigated this loss of data samples during the flight using the raw data provided by the manufacturer in json format and by correlating the raw data with the eye images.

We analyzed the accuracy between gaze point and target point from the videos recorded in different G values by calculating the distance between target point and gaze point distance. We used image processing methods (Figure 8) for this measurement. Initially, we did color transformation and removed noise in the image and applied adaptive threshold to find regions of interest as written in Algorithm 1. This image is given as input to Algorithm 2. We applied Hough transform to find both target and gaze circles in the preprocessed image. It returns Euclidean and Manhattan distance in units of pixels (Figure 9). To convert the pixel distance into centimeters, we measured the radius of the target circle in the photo, which was 2.2 cm. We measured the area of the contour around the target circle, and thus measured radius in pixels, which was 59. The whole conversion is done as follows:

$$Radius\ of\ target\ circle_{image} = 59\ pixels$$
$$Radius\ of\ target\ circle_{original} = 2.2\ cm$$
$$Radius\ of\ target\ circle_{image} = Radius\ of\ target\ circle_{original}$$
$$59\ pixels = 2.2\ cm$$
$$1\ pixel = (2.2/59)\ cm = 0.037\ cm$$
$$Euclidian\ distance_{in\ cm} = Euclidian\ distance_{in\ pixels} \times 0.037$$

Next, we measured the distance from the pilot's eye to the stimulus and converted the distance to visual angle. We compared the average error in accuracy in different G values in the two different aircraft (Figure 10). We undertook statistical analysis to find any significant difference in accuracy in different G values for both aircraft. We found a significant effect of conditions [$F(5,1416)=102.94$, $p<0.05$, $\eta^2=0.27$] for the Hawk aircraft. However, we did not find any significance difference between different G values in the Jaguar aircraft. A set of pairwise t-tests confirmed significant differences between errors in different G values at $p<0.05$, though there was no significance difference in errors in different timestamps of 1G.

function ImagePreprocessing(I)

Input: Input image of size M×N
Output: Preprocessed Image of size M×N
1 Convert the Input Image to gray scale.
2 Apply the Gaussian filter on Image obtained in Step-1
3 Apply the Adaptive thresholding on Image obtained in Step-2
4 Apply the Morphological transformation on Image obtained in Step-3
end

Algorithm 2. Gaze distance calculation.

function AverageGazeDistance (I)

Input: Input image of size M×N
Output: Average Gaze distance between Gaze circle and Target circle
1 Declare empty sets Circles = {} and Gaze_circle_list= {} and Target_circle_list = {} and Flag=0
2. Apply the Hough Transformation on Image and Append all the circle co-ordinates to the Circles List
3. if Circles = None
  then
    Go to **End**
4 for each $(X,Y,R)_{eachcircle} \in Circles_{List}$:
  if R < Gaze Circle Threshold
    then
      Append the (X, Y, R) to Gaze_circle_list
      Flag=1
  if R < Target Circle Threshold
    then
      Append the (X, Y, R) to Target_circle_list
end
5. if Flag = = 0 and Target_circle_list = = None
  then
    Go to **End**
6. $(x_1, y_1) \in Gaze\_circle\_list$ and $(x_2, y_2) \in Target\_circle\_list$
7 Calculate the Euclidean Distance (in pixels)
  Euclidean Distance = $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
8 Calculate the Manhattan Distance (in pixels)
  Manhattan Distance = $|x_2 - x_1| + |y_2 - y_1|$
End

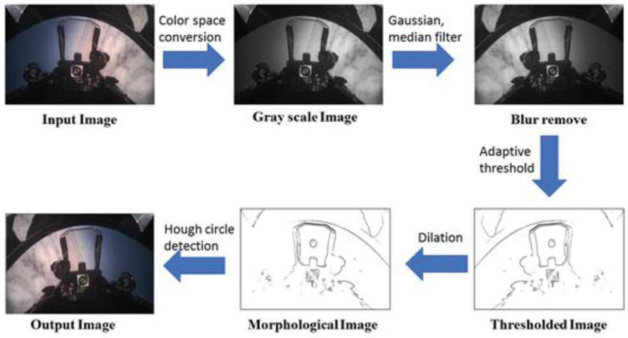


Figure 8. Different processing steps of the algorithm.

Figure 9. Evaluation procedure.

We also analyzed different ocular parameters like eye-gaze fixation and pupil dilation during different phases of flight. It may be noted from Figure 11 that the average fixation duration and fixation rate were both highest during inverted flight at −1G and either fixation duration or fixation rate was higher for 3G and 5G conditions compared to 1G condition.

In line with our error measurements of COTS eye-gaze trackers under various flight conditions, in the next section, we present our analysis on various failure modes of COTS

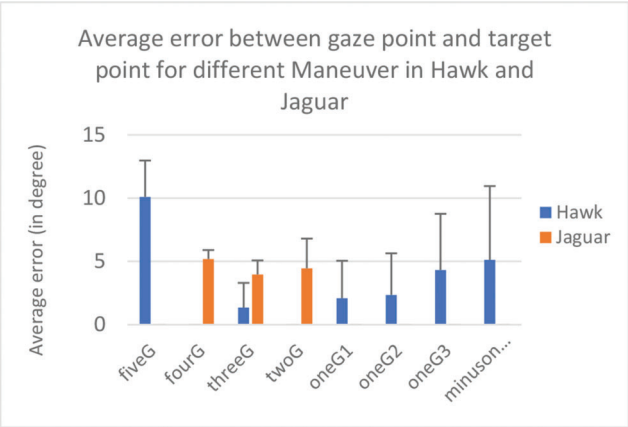


Figure 10. Average error in gaze accuracy.

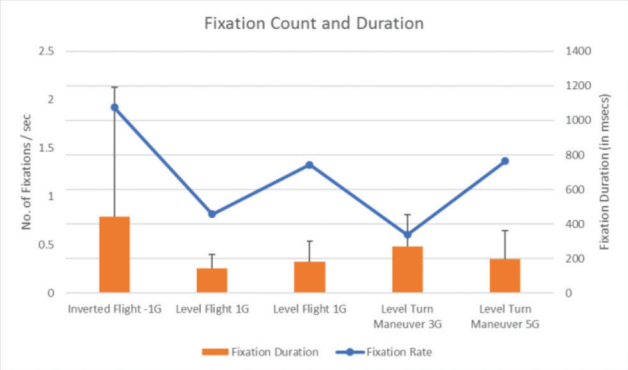


Figure 11. Fixations at different phases of flight.

eye-gaze trackers in actual fighter aircraft under flight conditions.

5. Analysis on Failure Modes

Further to the accuracy analysis, we analyzed the effect of high ambient illumination on the accuracy of eye-gaze tracking. It may be noted that the particular eye-tracking glasses were evaluated between 0 and 3000 lux (Tobii Pro Glasses 2 Product Description, 2019) according to technical specification. We observed that gaze points were not recorded for a significant flying time. To investigate the scenarios in which this loss of gaze data can happen, we performed a detailed analysis using the raw data obtained from the eye-gaze tracker. We discarded gyroscope and accelerometer data and retained the data points required for our investigation of lost gaze points.

At first, we synchronized the raw data stream and the eye camera stream in time scale since the eye camera stream starts off with an offset from raw data. This is achieved using the position time stamps provided in both data streams. We also find that the different frequencies of these two streams are a challenge for data synchronization. Hence, we considered the time duration between two successive frames of the eye camera stream and considered all the corresponding gaze data points recorded during that time window. Thus, the latter frame and these data points together form one pair of synchronized data points. Each time windowed raw data point may contain multiple gaze points. Every gaze point with its "gidx" contains a status code, *s*, which indicates the error associated with that data point, if any. The status code 0 indicates no error and any nonzero value of *s* indicates an error associated with the data point. We observed that all the gaze points with a nonzero status code are recorded as zeros for both *x* and *y* directions [0.0, 0.0]. The gaze points are provided in normalized values; hence the minimum gaze point is [0.0, 0.0] and the maximum is [1.0, 1.0].

We segmented the synchronized data points into two categories. The first category, *category1*, contains eye stream frames whose corresponding gaze points have zero status code. The second category, *category2*, contains those eye frames with all corresponding gaze points with nonzero status codes. There are frames whose data points have only a subset of gaze points containing zero status code. We did not consider these frames in our analysis as it brings uncertainty on eye image tagging. In other words, *category1* contains camera frames with proper gaze points and *category2* contains frames with no gaze points.

For Flight #1, we observed that out of 167,647 frames, only 57,111 frames fall under *category1* and 69,732 frames fall under *category2*. For Flight #2, we observed that out of 167,567 frames, only 81,911 frames fall under *category1* and 51,402 frames fall under*category2*.

Summarizing, 41.6% of the frames do not have any gaze points recorded during Flight #1 and for Flight #2, this stands at 30.7%. Further, if we just look at unsynchronized raw data, both flights recorded more than 51% of the gaze samples that are error-prone.

We visually inspected these flight recordings and we hypothesize two reasons for this data loss.

1. *Higher levels of illumination on eyes may affect the eye tracker resulting in no gaze estimation.*
2. *Limited field of view, especially in the vertical direction, renders the eye tracker with no gaze estimates when the user looks beyond the tracking range.*

We validated our hypothesis 1 using the eye images in the above mentioned two categories. We converted all eye images into grayscale and computed the average of all the pixel values for each image present in both *category1* and *category 2*. Figure 12 presents the histogram of image intensities for *category1* and *category2* for Flight #1. Figure 13 presents the same for Flight #2.

Figure 12a indicates that 93% of the images under *category1* have an average pixel value less than 131. But, *category2* contains 42% with average intensity higher than 131. In other words, only 7% of the images with intensity

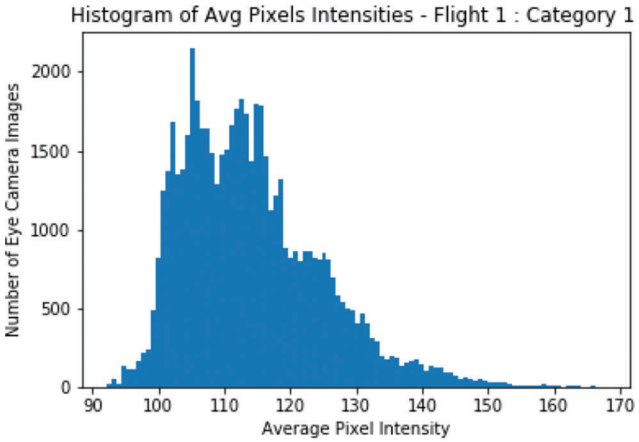


Figure 12. Histogram of image intensities for Flight 1: (a) *category1*; (b) *category2*.

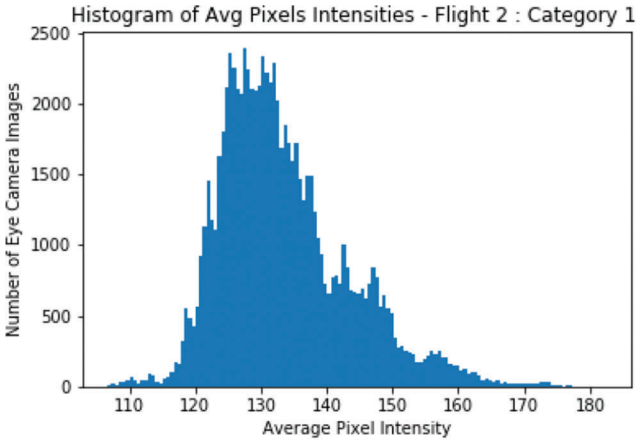


Figure 13. Histogram of image intensities for Flight 2: (a) *category1*; (b) *category2*.

higher than 131 have proper gaze points and over 42% of images which do not have gaze points have intensity higher than 131.

A similar phenomenon can be observed in Flight #2, shown in Figure 13. Around 42% in *category2* have intensity higher than 150, while *category1* contains 94% of the images with intensity less than 150. In this case too, only 6% of the images with intensity higher than 150 have proper gaze points and over 42% of images which do not have gaze points have intensity higher than 150. This indicates that images with higher illumination, precisely above 131 in Flight #1 and above 150 in Flight #2, have low probability to obtain accurate gaze estimates.

While this evidence supports our hypothesis 1 partially, we observed that there is overlap in the histograms plotted in Figures 12 and 13. Hence, we could not identify a clear average image intensity threshold to identify all the failure modes of eye-gaze estimation.

We further investigated the data points in *category2* to understand the 58% of the data points which have lower image intensities than the above mentioned thresholds for each flight using our hypothesis 2.

During our visual inspection of first-person video recorded using eye tracker, we observed that the pilot looks down for various activities like looking at the information displayed in the MFDs. During such scenarios, we observed that gaze points were not recorded.

Extending our hypothesis 2, we assumed that the pilot must be looking at a position closer to the extreme tracking positions (beyond which eye tracker cannot track), just before or after the eye tracker fails to provide gaze estimates. Since we observed that the gaze estimates are lost for a sequence of eye image frames, we clustered the data points in *category2* based on their gidx's. If a sequence of data points under *category2* have successive gidx's, then all those points are collectively considered as a single cluster. Hence, each cluster can contain one data point or several data points.

Hence, we analyzed three preceding and subsequent data points adjacent to each cluster, which we refer to as boundary data points. We looked for boundary data points with gaze values greater than 0.8 and less than 0.2 (in both x and y). If any of the boundary data points satisfy the above criterion, then we may infer that the loss of gaze points is due to the pilot looking beyond the tracking range of the eye tracker.

For Flight #1, we obtained 12,178 clusters for 69,732 data points. For these clusters, 11,865 (97.43%) clusters

have boundary points that satisfy the above criterion. To understand image intensities for these data points, we obtained the image intensities for boundary data points that satisfy the above criterion. We observed that these image intensities lie in the range of (96, 145). This is clearly in the overlap range identified between Figure 12a and Figure 12b.

Similarly, for Flight #2, we obtained 8646 clusters for 51,402 data points. For these clusters, 8408 (97.24%) clusters have boundary points that satisfy the above criterion. Interestingly here as well, we observed that the image intensities for the above points lie in the range of (117, 164), which is the range of overlap identified in Figure 13a and Figure 13b.

Thus, we infer that this eye-gaze tracker could not identify beyond a certain illumination level or if the user is looking beyond its tracking range. We should note that the pilots in both flights were performing their assigned tasks during the flight and maintained their natural behavior. This indicates that the tracking range offered by the eye tracker used in this study falls short for military aviation environments.

Hence, using our two hypotheses, we explained the failure modes of an eye-gaze tracker in an aviation environment. We further add that, while COTS eye trackers may be used in real aviation environments, researchers and practitioners should keep in mind both the horizontal and vertical tracking range of the eye tracker and its robustness to external illumination as there is a high chance that the illumination varies rapidly at high altitudes and in high-speed maneuvers.

6. Theory and Design of HMDS

With this understanding of how pilots' gaze patterns and the behavior of eye-gaze trackers vary over various critical flying tasks, we consider adaptive displays which facilitate eye-gaze-based interactions as a solution to reduce the cognitive load of the pilot and the error in interaction (Shree et al., 2018). This section describes design and evaluation of such a system controlled by head and eye-gaze movement.

Gaze Direction Vectors

We used a COTS wearable eye-gaze tracker (Tobii Pro Glasses 2) to capture gaze direction unit vectors for each eye. These vectors were measured with the center of the respective pupil as the origin. In subsequent sections, we term the left and right eye's gaze direction vectors as *eyeL* and *eyeR* with dimensions 3×1. Wearable eye trackers provide gaze information with respect to a given head position. Figure 14 illustrates this with two instances of a user gazing at two different points wearing an eye-gaze tracker. In the first instance, the user looks straight at point A. In the second, the user turns their head towards the right by α degrees and looks straight at point B. The eye-gaze vectors from the eye tracker would be the same in these two cases even though the user is looking at two different points in space.

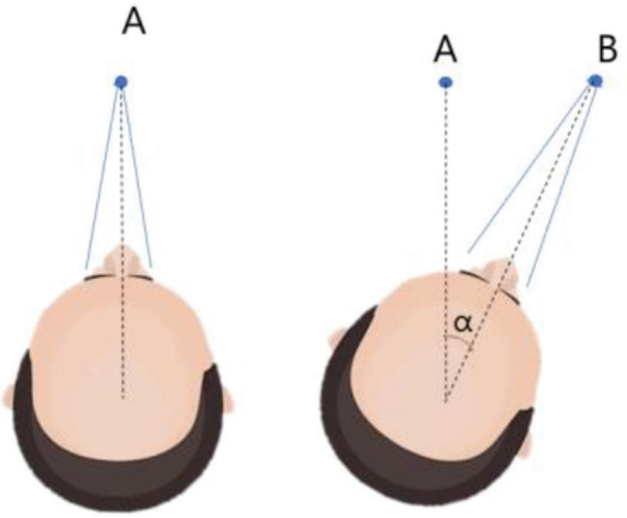


Figure 14. Illustration of gaze direction vectors along with head movement.

Head Orientation

Tobii Pro Glasses 2 has in-built MEMS (microelectromechanical system) accelerometer and gyroscope. These MEMS sensors are prone to noise and absence of an in-built magnetometer does not guarantee an accurate measurement of head yaw (LaValle et al., 2014). Hence, we used a nine-axis IMU to measure head yaw, pitch, and roll and it is placed right above the user's head. We considered the initial position of the user's head as the reference head position. The three mutually perpendicular axes passing through the center of the IMU at this position were considered as the reference coordinate axes. Figure 15 illustrates yaw, pitch, and roll with respect to the user's head and these are the orientations about axes z, y, and x, respectively. In subsequent sections, we term the yaw, pitch, and roll of the head as α, β, and γ, respectively.

Head-Compensated Gaze Vectors

Every sample of *eyeL* and *eyeR* obtained along with a given head orientation was transformed to the reference coordinate axes using the following intrinsic three-dimensional transformation (Goldstein et al., 2002). We termed the head-compensated gaze vectors as $eyeL_{hc}$ and $eyeR_{hc}$, with dimensions 3×1. Here, we assumed that both eyes are positioned equidistant from the origin of the reference coordinate axes. The above framework provided unique $eyeL_{hc}$ and $eyeR_{hc}$ for a given point in space. These head-

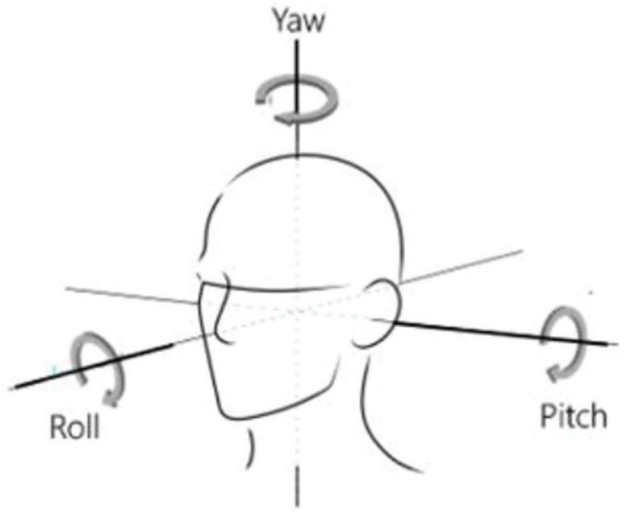


Figure 15. Yaw, pitch, and roll for a head movement.

compensated eye vectors, $eyeL_{hc}$ and $eyeR_{hc}$ were cascaded into a single vector eye_{hc}, with dimensions 6 × 1.

$$T_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (1)$$

$$T_y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \quad (2)$$

$$T_x(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix} \quad (3)$$

$$T = T_z(\alpha)T_y(\beta)T_x(\gamma) \quad (4)$$

$$eyeL_{hc} = TeyeL$$

$$eyeR_{hc} = TeyeR$$

We analyzed gaze direction unit vectors for various screen coordinates and inferred that the relationship between components of these vectors and screen positions was not linear, which is consistent with earlier findings (Shree et al., 2018). Hence, instead of a linear formulation, we used a nine-point calibration routine to obtain eye_{hc} at different locations on screen and used a feedforward neural network to learn the mapping function.

Calibration and Operation

We used an 80" projected display of resolution 800 × 600 and users were requested to sit at 3.2 m from it. The setup can be viewed in supplementary material 2. Users were asked to wear eye-tracking glasses and an IMU-attached cap (Figure 16). We displayed nine squares of size 90 × 90 px as calibration markers on screen one after another. Pfeuffer et al. (2013) reported limitations of using static calibration markers. To overcome such limitations, we provided visual feedback in response to the user's focus on the square. The size of the square reduced continuously when the user focused on it until it reached a minimum size of 10 × 10 px. We measured standard deviation of gaze vector components to measure the user's focus on the square. In the case where the user looked away, standard deviation increased and if it was greater than the design threshold, the square regained the original size. This method allows the user to stop and resume the calibration at his/her will.

Figure 16. Participant wearing eye-tracking glasses and a cap with IR reflective markers and IMU.

When the square reached the minimum size, eye_{hc} vectors are recorded. We chose these nine points in such a way that they span across the screen. We used Tensorflow.NET and Keras.NET, the .NET standard bindings of widely used *tensorflow* and *keras* python packages, for building and training the neural network. A two-hidden-layer neural network was trained with mean squared error loss function and Adam optimizer. We used training loss and coefficient of determination to determine the termination condition of training the network. In addition to that, these parameters were useful in preventing overfitting. The training started once the eye_{hc} vectors were obtained for all nine points and the average time for neural network training was observed

to be 6 seconds. Once the network was trained, predictions of the neural network were used to move the screen cursor.

We performed time-window average of 0.2 seconds on input gaze vectors and output predictions to achieve a smooth cursor movement. In addition to that, we also used the following measures to keep the cursor less jittery.

- **Pixel threshold.** We measured the Euclidian distance between successive predictions from the neural network. We updated the cursor position only if that distance was above a certain pixel threshold.
- **Angle step threshold.** As it is natural to have small head movements, there will be a continuous change in orientation values. In addition to this, IMU has a root-mean-square error of 2° and takes 0.2–0.5 seconds to converge to an accurate value. Hence, we updated the head orientation only if the incoming orientation value differed from the current value by an angle step threshold.

These two design parameters affect both task completion time and jitter in cursor movement. If we set these thresholds too high, small cursor movements cannot be made, and if we set these too low, the user might be subjected to irritation with unintended cursor movements. We determined these thresholds based on the results we obtained from preliminary pilot studies. Even though the previously mentioned framework provides direct mapping of eye_{hc} to screen coordinates, error from the IMU affects predictions from the neural network, which results in cursor offset. Participants are able to correct this error by moving their heads while keeping their gaze at the same point on screen.

7. User Study on HMDS

We conducted a user study to compare two interaction modalities, the existing joystick-based TDS and the proposed multimodal head and eye-gaze interface (MMHE). In this section, we describe the flight simulator and design of the study involving various sensors.

Flight Simulator Setup

We designed a flight simulator to conduct the user study in a dual-task setting. Using our setup, participants undertook standard flying tasks in parallel with representative pointing and selection tasks. This setup allowed us to measure not only pointing and selection times, but also the total response time, consisting of the time required to switch from primary flying to secondary mission control tasks. We designed our study to emulate a HMDS where information is projected on to the visual field. Participants needed to interact with that information along with the primary flying task. The flight simulator was projected onto an 80" display.

Third-party flight simulator YSFlight with data logging feature was configured for this study as an F/18 E/F Super Hornet aircraft. The flight simulator was configured with a Thrustmaster Warthog HOTAS (Hands On Throttle and Stick). Both flight simulator and secondary pointing tasks were run on an Intel Pentium CPU G3220@3GHz computer running Windows 7 operating system with 4GB RAM and Nvidia GeForce 210 graphics card.

Head Tracking Using OptiTrack

During the development, we observed that IMU values drifted from actual values which may result in erroneous condition. To study this in detail, we used a COTS IR-based motion capture system (OptiTrack system) to obtain head orientation. We placed five retro-reflective markers onto the same cap where the IMU was placed. We used five Flex 13 cameras to obtain head orientation. We did not use head orientation values obtained from OptiTrack as part of our proposed interface; rather, we investigated the correlation between the head orientation values obtained from IMU with head orientation from OptiTrack. This setup can also act as a testbed and allows us to compare any other head orientation measuring technique apart from IMU in the future, enabling us to choose the most accurate head tracking system to integrate with our gaze interface. Since the sampling rate of IMU and OptiTrack is different, we performed time sampling of 1 second to compute the average value. These average values were used to compute correlation.

Flying Task

A map was configured with a straight line drawn in the middle. Participants were instructed to fly between 1000 and 2500 feet along the straight line. The secondary task was initiated after the flight reached the designated flight envelope of 1000 and 2500 feet.

Secondary Task

We designed a pointing and selection task similar to ISO 9241-9 (Figure 17). This task was overlapped onto the flight simulator and participants undertook this task while flying. The task was to click a button at the center of the screen followed by clicking a randomly generated red color target button. The time between these two clicks was measured as the selection time.

We considered three widths (W) of 70 px, 80 px, and 90 px (corresponding to 12 cm, 14 cm, and 16 cm) for target buttons and three distances (D) between the center button and the target buttons of 200 px, 220 px, and 240 px. This leads to a total of nine ID cases. The order of these ID cases and interaction modalities was randomized for all participants. The above-mentioned target widths subtended a visual angle of 2.14°, 2.5°, and 2.86°, respectively.

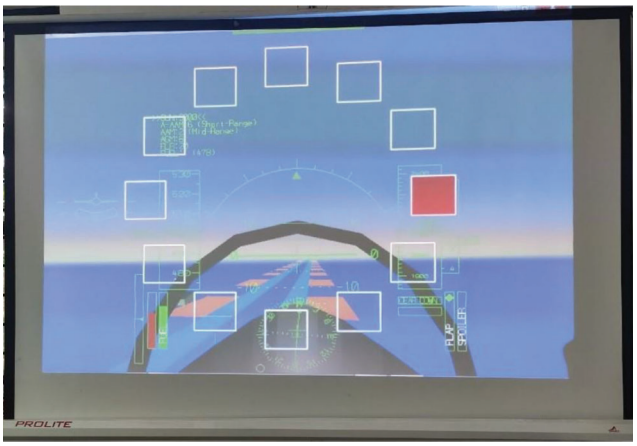


Figure 17. ISO pointing task overlapped onto the primary flying task.

Table 1
Summary of interaction metrics.

Metric	Joystick	MMHE
Movement time, MT (ms)	4456 (731)	3017 (909)
Throughput, TP (bits/s)	0.434 (0.04)	0.686 (0.20)
TLX score	45.63 (15.8)	37.92 (15.49)
SUS score	64.68 (14.9)	73.44 (13.37)

Table 2
Summary of flying performance.

Metric	Joystick	MMHE
Deviation from path	486.8 (591)	375.2 (357.1)
Altitude deviation	199.4 (46.4)	199.4 (39.7)
Average flight distance (m)	56,564	53,313

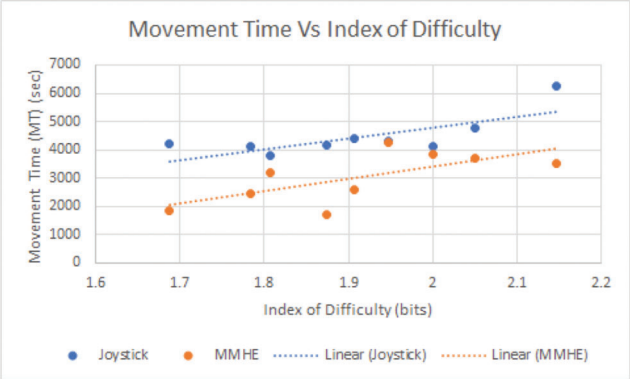


Figure 18. Movement time versus ID for joystick and MMHE.

In the case of joystick, participants used the trackball on the throttle for both pointing and selection whereas in the case of MMHE, participants pointed using their eye gaze or/and head and selected using the button on the throttle. The time taken by each participant for the study was recorded since the inception of take-off. The task was considered complete when participants performed all 18 clicks using a given modality or when they completed 6 minutes since take-off, whichever was earlier.

We collected data from 8 participants (7 male, 1 female) aged between 23 and 28 years (mean = 25, SD = 1.51). Each participant was instructed to consider flying as the primary task and perform the secondary task only when he/she felt their flight was satisfying the flying task instructions. Each participant performed the task 2 times for each ID case and hence a total of 18 clicks in each mode of interaction. Movement time (MT) was measured as the average of selection times across all participants for a given ID. ID and throughput (TP) were calculated based on the following formulae:

$$ID = \log_2\left(\frac{D}{W} + 1\right) \quad TP = \frac{ID}{MT}$$

All participants were asked to familiarize themselves with the interface and the actual trial was conducted only after they felt confident in using the system. In the case of MMHE, participants were briefed about the head movements they could perform to look as well as to correct the offset. After each participant completed his/her trial, subjective feedback was collected using NASA TLX for cognitive load and SUS questionnaire for subjective preference. A demonstration of the system can be viewed in supplementary material 2.

Results

We summarize both quantitative and qualitative metrics of interaction in Table 1. We measured mean values of metrics for all participants and calculated the standard deviation, which is given in parentheses.

We analyzed movement times for all IDs for both joystick and MMHE modalities. The average MT for joystick and MMHE modalities is 4.5 and 3.0 seconds, respectively. Figure 18 shows the MT for all ID cases and the dashed line indicates the trend line for each modality. We undertook a paired *t*-test (*t*: 4.31, $p = 0.001$, Cohen's *d*: 1.44) for MT and found that participants took significantly less time in using MMHE than joystick.

The average TLX scores for joystick and MMHE are 45.63 and 37.92, respectively. A paired *t*-test (*t*: 2.31, $p = 0.027$, Cohen's *d*: 0.82) for TLX score indicates that the perceived cognitive load in the MMHE case is significantly lower than in the joystick case. Even though the average SUS score for MMHE is higher than for joystick, a paired *t*-test for SUS scores indicates that the subjective preference between joystick and MMHE is not significantly different.

Discussion

In addition to the task completion metrics and qualitative feedback, we analyzed cursor movement efficiency metrics

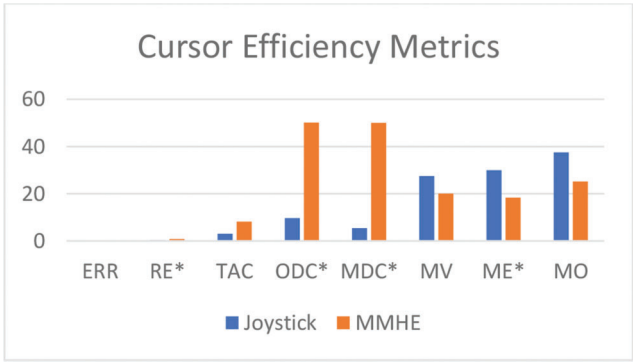


Figure 19. Comparing cursor efficiency metrics.

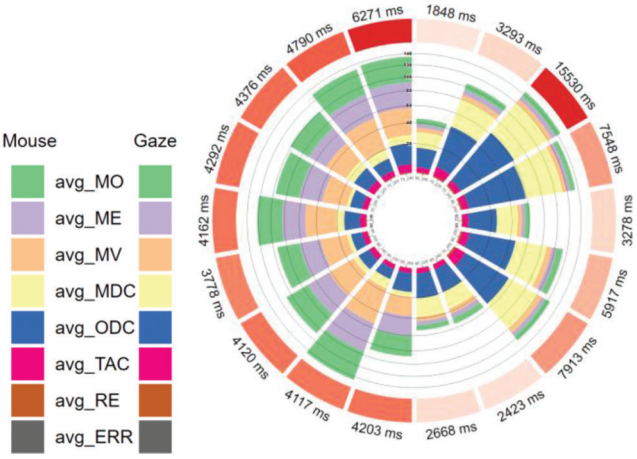


Figure 20. Radial stacked bar chart.

for both modalities (Figure 19) which were introduced in Section 3. We used an asterisk on the graph for parameters that were significantly different at $p < 0.05$ in a paired t-test. The cursor efficiency metrics computation took the entire trajectory of the mouse from the center button click until the target button click. The cursor was moving along with eye movement in this dual-task setting while using MMHE unlike the joystick case.

We observed participants assessing their flight control by observing the altimeter and relative position with respect to the central path after clicking the center button and before clicking the target button. The task timer is turned on during this duration. A significantly higher ODC (t: 4.38, $p = 0.002$, Cohen's d: 1.55) and MDC (t: 4.73, $p = 0.001$, Cohen's d: 1.68) and higher TAC in MMHE than in joystick can be explained by this observation. The average RE in MMHE and joystick is 0.85 and 0.34, respectively. MMHE has lower values of metrics that look at the variability of the movement (MV, MO), and significantly lower (t: -2.01, $p = 0.04$, Cohen's d: -0.71) than joystick in terms of ME.

We further analyzed the data with radial stacked bar chart (Figure 20) to obtain better insight about the relation between the above-mentioned metrics and various ID cases. The left and right sides of the radial stacked bar chart represent MMHE and joystick, respectively. We can see from the figure that the extent of green, violet, and orange representing average movement offset (avg_MO), average movement error (avg_ME), and average movement variability (avg_MV) for all the ID cases on the left-hand side is smaller than on the right-hand side of the chart.

We compared participants' flying performance while performing the task with both interaction modalities. Table 2 summarizes the three metrics that represent flying performance. Participants had to fly longer to complete the task when joystick was used than MMHE. The deviation from central path was also higher while using joystick than MMHE. The altitude deviation was not significantly different between two interaction cases.

We measured head orientation using both OptiTrack and IMU sensors. Out of eight participants, three participants' head orientation data measured from OptiTrack had high error per marker and missing data, and hence we could not consider those data. We measured correlation between the values from IMU and OptiTrack for the remaining five participants as reported in Table 3. We observed high correlation between IMU and OptiTrack for both yaw (0.85) and pitch (0.77) measurements. We observed positive, but low correlation for roll measurements (0.45). While using MMHE, we observed that participants performed offset correction using their head movements when targets appeared in the upper quarter of the task region. This might be attributed to a lower correlation of pitch when compared to yaw where pitch movement was required to look at the target.

Table 3
Correlation of head orientation between OptiTrack and IMU.

Participant	Yaw	Pitch	Roll
1	0.83	0.82	0.23
2	0.92	0.72	0.76
3	0.93	0.67	0.38
4	0.85	0.78	0.44
5	0.75	0.88	0.43
Average	0.85	0.77	0.45

Thus, we developed and evaluated a multimodal interface controlled by eye gaze and head movement on a simulated HMDS. In the next section, we describe screen-mounted eye tracker development which can be utilized to operate MFDs in military aviation.

8. Screen-Mounted Eye Tracker for MFDs

In this section, we describe three different eye-tracking systems which we compared through user studies. Initially we describe the different algorithms used for estimating gaze followed by two user studies.

Histogram of Oriented Gradients-Based Gaze Tracking System

We used a pre-trained facial landmark detector with iBUG 300-W dataset (iBUG, 2019), which uses classic histogram of oriented gradients (HoG) feature combined with a linear classifier to detect facial landmarks (Rosebrock, 2019). In comparison, Haar cascades are a fast way to detect an object but often detect more false positives (Dalal & Triggs, 2005). HoG features can describe shape better than Haar features, and Haar features can describe shading better than HoG features. In this case the shape is more important as we need the landmarks of the face; hence HoG features produced a better result. After detecting the eye region, we detected pupil location by selecting the smallest rectangle possible in the eye region where the pupil can exist. After retrieving pupil locations, we calculated the eye aspect ratio (EAR) (Cech & Soukupova, 2016). We note that the EAR changes with respect to the distance between the user and the camera. We modified the EAR calculation formula by using the distance between the two eyes as denominator.

Webgazer.js

We implemented a second system using webgazer.js (Agarwal et al., 2019; Papoutsaki et al., 2016) to compare the performance of the proposed system. Webgazer.js runs entirely in the client browser. It requires a bounding box that includes the pixels from the live video feed that corresponds to the detected eyes of the user. This system uses three external libraries (clmtracker, js_objectdetect, and tracking.js) to detect face and eyes. It has methods for controlling the operation which allow us to start and stop it. We took the mean of the last 30 points from webgazer.js for better target prediction and accuracy of the system. We also calculated the mean value during this time to predict the gaze location on a webpage.

Intelligent System

We have developed a gaze block estimator which maps a user's eye movements to nine screen blocks using Open Face (Baltrusaitis et al., 2018) toolkit. Since OpenFace (Figure 21) was reported to have a cross-dataset validation error of 6° for gaze point estimation, we designed a calibration routine which uses the gaze vector data from OpenFace and maps the user's eye movements to screen blocks, instead of screen pixels. We divided the screen into nine blocks of equal area. We designed a smooth pursuit-based calibration routine where a marker traverses across all the nine blocks and the user was asked to follow the marker's movement. While the user followed the moving marker, the corresponding gaze vectors from OpenFace

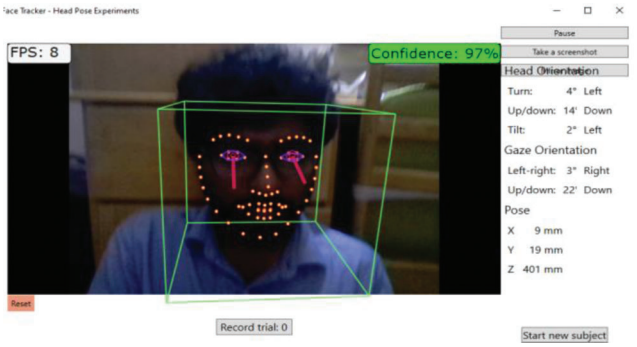


Figure 21. Screenshot from OpenFace face tracker.

were recorded and stored with the respective block number as the label. Once the marker completes its path, a neural network is trained to map these gaze vectors to nine blocks of the screen. For this classification task, we used a two-hidden-layer network with 256 and 128 neurons, respectively, with cross-entropy loss function and with Adam optimizer. We used 70% of the data we recorded during calibration for training, 15% for validation, and the rest for testing. On an i7 processor computer, we observed that each epoch takes around 0.8 seconds and we trained the network until the test accuracy reached 90%.

User Study

We undertook a user study to compare different eye-tracker implementations in different lighting conditions and compared them with a COTS screen-mounted eye-gaze tracker.

Participants

We collected data from 9 participants (8 male, 1 female). All participants were recruited from our university. They did not have any visual or motor impairment.

Material

The user trial was conducted on a Microsoft Surface Pro tablet powered by a dual-core processor with 8 GB RAM and running Microsoft Windows 10 operating system. The surface has a 5 MP camera, which was used to estimate gaze direction.

Design

We wanted to use the eye tracker to operate a graphical user interface with limited number of screen elements. Hence instead of traditional precision and accuracy measurements, we calculated the pointing and selection times for a set of fixed positions on screen.

We created a user application in which we divided the screen into nine blocks, with one of the blocks randomly

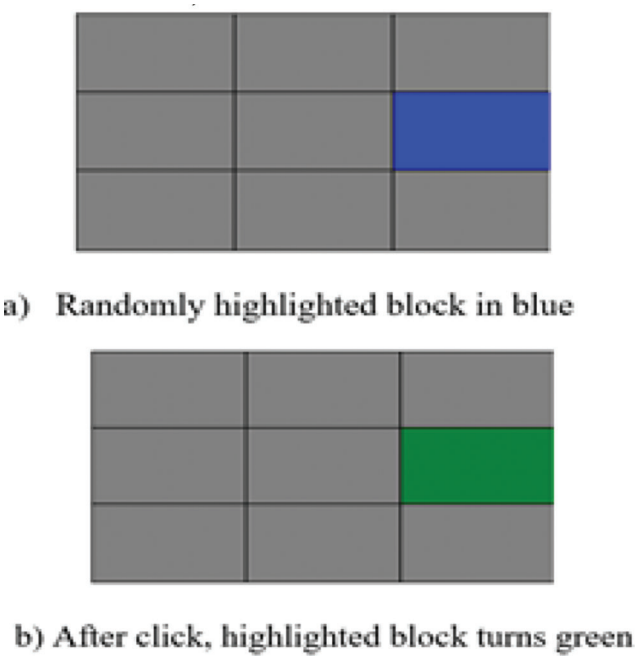


Figure 22. Pointing task application.

highlighted with blue color as shown in Figure 22a. If the user clicks on the blue block, it turns green as shown in Figure 22b and a different block is highlighted. If the user is unable to click on the highlighted block within 10 seconds, some other block turns randomly to blue. Using this interface, we calculated the response time by measuring the time difference between the appearance of a highlighted block and its selection. Users selected the target using the left mouse button.

The trial was performed twice: once in the laboratory with lux meter reading of 180–200 lux and the other in outdoor conditions with lux meter reading between 1800 and 3000 lux.

The trial consisted of four eye-tracking implementations:

- HoG-based bespoke screen-mounted gaze tracker.
- Webgazer.js-based screen-mounted gaze tracker.
- OpenFace-based intelligent screen-mounted gaze tracker.
- Tobii PCEye Mini Eye Tracker (referred to as COTS tracker).

For all trial conditions, we conducted the trial using the same user application discussed before. The order of conditions was randomized to minimize practice or learning effect. Each participant undertook all trial conditions.

Results

We recorded 322 pointing tasks inside and 270 pointing tasks outside. We measured the time difference between onset of a target and its correct selection. We removed outliers by identifying points greater than outer fence. We removed only one point from the selection times recorded from the bespoke eye-gaze tracker while 12 data points were found to have values higher than outer fence for the COTS eye-gaze tracker.

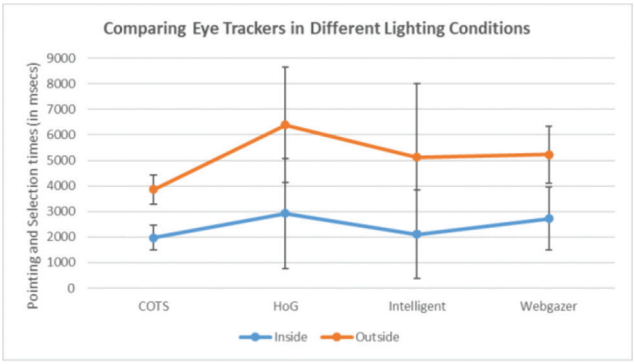


Figure 23. Comparing average response time among different eye trackers.

Figure 23 presents the average selection times and standard deviations. Participants took the shortest time to select the target using the COTS tracker. We undertook a 2 × 2 unbalance factorial regression-based ANOVA (type of eye gaze trackers × lighting conditions) on the response times.

We found

- significant main effect of type of eye-gaze tracker $F(3, 567) = 15.44$, $p < 0.01$,
- significant main effect of lighting condition $F(1, 567) = 4.05$, $p < 0.05$, and
- significant interaction effect of type of eye-gaze tracker and lighting condition $F(3, 574) = 3.45$, $p < 0.05$.

Then we undertook two one-way ANOVAs for each lighting condition and found significant main effect of eye-gaze tracker implementations:

- Inside room $F(3, 318) = 8.43$, $p < 0.01$.
- Outside room $F(3, 266) = 11.31$, $p < 0.01$.

Finally, a pair of unequal variance *t*-tests did not find any significant difference between the COTS tracker and our intelligent eye-gaze tracker implementation under the inside condition at $p < 0.05$, although the difference in response times between the intelligent system and the COTS tracker was significant under the outside condition at $p < 0.05$.

9. Discussion

Even though numerous studies were conducted earlier evaluating the utility of eye-gaze trackers in military aviation, we conducted multiple user studies in actual flying conditions. Earlier, Wilson et al. (1994) collected

data for air to ground dive attack maneuvers but did not report ocular parameters beyond rate and duration of eye blinks. Our user studies could help to understand the effectiveness of eye-gaze-controlled interface in transport aircraft in terms of pointing and selection tasks. Further, we performed another study to investigate the accuracy of COTS eye trackers under various G-scenarios which occur normally as a part of military flight operations. Our analysis shows that an existing COTS sensor can track eye gaze within 4° of visual angle up to +3G and accuracy reduces to 9.5° of visual angle at +5G. This measurement can be used to design future displays for eye-gaze-controlled HMDS. However, it may be noted that this paper reports results from only one pilot who has not used eye-gaze-controlled interfaces before, and future studies will collect data from more pilots.

Using the data from our studies further, we studied and identified the probable conditions where COTS eye trackers may fall short of expectations in terms of military aviation requirements. We identified that the COTS eye tracker that we considered had less of a vertical tracking field than what is necessary in common military aviation operations. We also found that high natural illumination externally may cause failure of IR-based eye-tracking systems to provide gaze estimations. In our current analysis, the illumination levels vary between the two flights under consideration. We may need to analyze more flight data to identify a definitive illumination level beyond which the COTS eye tracker may fail. In this direction, we attempted to develop eye trackers in natural illumination conditions using a screen-mounted COTS camera. We utilized an existing state-of-the-art deep learning-based eye-gaze estimation method to develop an eye-gaze-controlled interface. Our evaluation of this system along with other screen-mounted approaches indicates that these systems can be utilized to perform eye-gaze estimation under natural sun light, since they do not rely on IR illumination. The reported task times in Figure 23 indicate that the COTS eye tracker performed faster gaze estimation, but this may be due to the fact that the COTS eye tracker is built on dedicated optimized hardware, whereas other systems are evaluated on consumer-level laptops. In the future, we will focus on developing gaze point level estimation using deep learning techniques, instead of gaze block estimation from camera images.

10. Conclusions

This paper reports two user studies on analyzing the accuracy of eye-gaze-controlled interface using COTS eye-gaze trackers in a military aviation environment. The main findings of our study are the following.

1. Pilots can undertake representative pointing and selection tasks at an average duration of less than 2 seconds and less than 5% error rate using eye-gaze-controlled interface.
2. Accuracy of commercial eye-gaze-tracking glasses reduces when the constant load factor of the aircraft is more than +3G or less than 0G.

However, our studies involved only a limited set of pilots who were not exposed to eye-gaze-controlled interfaces earlier.

We furthermore described a new multimodal head and eye-gaze movement-controlled HMDS and compared the performance of the system with that of a traditional joystick-based TDS in a flight simulator. From our user study, we observed that participants took significantly less time to interact with the targets and perceived significantly less cognitive load using the proposed interface than with the existing system. We observed that the cursor movement variation metrics are lower for the MMHE than for the existing joystick system. In addition to that, participants deviated less and completed the task in shorter distance using our proposed system. These results motivate us to design a multimodal head and gaze interactive HMDS. We plan to set up a testbed for head-tracking systems which we plan to develop in the future to integrate with our proposed framework.

We also described the evaluation of our screen-mounted eye-gaze tracking system against a COTS IR-based eye-gaze tracker and other screen-mounted camera-based eye-gaze tracking approaches. We show that interaction with our system allowed users to complete the pointing and selection tasks faster than with any other screen-mounted approaches and we also demonstrate the robustness of screen-mounted approaches to external illumination conditions. Our future work includes the development of person-independent eye-gaze-point estimator systems using deep learning techniques.

Acknowledgments

The authors would like to thank all the participants who took part in the user studies.

References

Abel, G. J., & Sander, N. (2014). Quantifying global international migration flows. *Science*, *343*(6178), 1520–1522. http://dx.doi.org/10.1126/science.1248676

Adelstein, B. D., Beutter, B. R., Kaiser, M. K., McCann, R. S., Stone, L. S., Anderson, M. R., Renema, F., & Paloski, W. H. (2008). *Influence of combined whole-body vibration plus G-loading on visual performance*. Report to the NASA Human Research Program.

Agarwal, A., JeevithaShree, D. V., Saluja, K. S., Sahay, A., Mounika, P., Sahu, A., & Biswas, P. (2019). Comparing two webcam-based eye gaze trackers for users with severe speech and motor impairment. In *Research into design for a connected world* (pp. 641–652). Springer.

Babu, M. D., JeevithaShree, D. V., Prabhakar, G., & Biswas, P. (2019). Using eye gaze tracker to automatically estimate pilots' cognitive load.

In *50th International Symposium of the Society for Flight Test Engineers (SFTE)*. http://dx.doi.org/10.16910/jemr.12.3.3

Baltrusaitis, T., Zadeh, A., Lim, Y. C., & Morency, L. P. (2018, May). OpenFace 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)* (pp. 59–66). IEEE. http://dx.doi.org/10.1109/FG.2018.00019

Biswas, P. (2016). *Using eye gaze controlled interfaces in automotive environments*. Springer.

Biswas, P., & Langdon, P. (2013). A new interaction technique involving eye gaze tracker and scanning system. *ACM International Conference Proceeding Series* (pp. 67–70). http://dx.doi.org/10.1145/2509315.2509322

Borah, J. (1995). *Investigation of eye and head controlled cursor positioning techniques*. Air Force Materiel Command Report, AL/CF-SR-1995-0018.

Calhoun, G. L., & Janson, W. P. (1991). *Eye line-of-sight control compared to manual selection of discrete switches*. Armstrong Laboratory Report AL-TR-1991-0015, Wright-Patterson Air Force Base, Ohio, USA.

Cech, J., & Soukupova, T. (2016). Real-time eye blink detection using facial landmarks. *21st Computer Vision Winter Workshop* (pp. 1–8).

Chernoff, H. (1973). The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, *68*(342), 361–368.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, USA (Vol. 1, pp. 886–893). http://dx.doi.org/10.1109/CVPR.2005.177

De Reus, A. J. C., Zon, R., & Ouwerkerk, R. (2012). Exploring the use of an eye tracker in a helmet mounted display. In *Avionics Europe Conference & Exhibition, Munich, Germany*, March 21–22, 2012.

Elbit Systems. (2020). Display and Sight Helmet System (DASH). Retrieved August 14, 2020, from https://elbitsystems.com/product/display-and-sight-helmet-system-dash/

Eurofighter. (2017). Flight manual for the package Eurofighter Typhoon Professional. Retrieved from https://www.afs-design.de/pdf/AFS_EFpro_English.pdf

Fitts, P. M., Jones, R. E., & Milton, J. L. (1950). Eye movements of aircraft pilots during instrument-landing approaches. *Aeronautical Engineering Revi*ew, *9*, 24–29.

Goldstein, H., Poole, C., & Safko, J. (2002). *Classical mechanics*. Addison Wesley.

iBUG 300-W dataset. (2019). https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/

LaValle, S. M., Yershova, A., Katsev, M., & Antonov, M. (2014, May). Head tracking for the Oculus Rift. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 187–194). IEEE.

MacKenzie, I. S., Kauppinen, T., & Silfverberg, M. (2001). Accuracy measures for evaluating computer pointing devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 9–16). ACM.

Papoutsaki, A., Sangkloy, P., Laskey, J., Daskalova, N., Huang, J., & Hays, J. (2016, January). Webgazer: Scalable webcam eye tracking using user interactions. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence—IJCAI 2016*.

Pfeuffer, K., Vidal, M., Turner, J., Bulling, A., & Gellersen, H. (2013, October). Pursuit calibration: Making gaze calibration less tedious and more flexible. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (pp. 261–270). ACM.

Rosebrock, A. (2019). Facial landmarks with dlib, OpenCV, and Python. Retrieved March 6, 2019, from https://www.pyimagesearch.com/2017/04/03/faciallandmarks-dlib-opencv-python/

Rudi, D., Kiefer, P., Giannopoulos, I., & Raubal, M. (2019). Gaze-based interactions in the cockpit of the future: A survey. *Journal on Multimodal User Interfaces*, 1–24.

Shree, D. V., Murthy, L. R. D., Saluja, K. S., & Biswas, P. (2018). Operating different displays in military fast jets using eye gaze tracker. *Journal of Aviation Technology and Engineering*, *8*(1), 31. http://dx.doi.org/10.7771/2159-6670.1184

Smyth, C. C. (1997). U.S. Patent No. 5,689,619. Washington, DC: U.S. Patent and Trademark Office.

Stasko, J., & Zhang, E. (2000). Focus+ context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *IEEE Symposium on Information Visualization 2000. INFOVIS 2000* (pp. 57–65). IEEE.

Striker® II Digital Helmet-Mounted Display. (2020). Retrieved August 14, 2020, from https://www.baesystems.com/en/product/striker-ii-digital-helmet-mounted-display

Thomas, P., Biswas, P., & Langdon, P. (2015). State-of-the-art and future concepts for interaction in aircraft cockpits. In *Lecture notes in computer science* (Vol. 9176, pp. 538–549). Springer. http://dx.doi.org/10.1007/978-3-319-20681-3_51

Tobii PCEye Mini Eye Tracker. (2018). Retrieved August 31, 2018, from https://www.tobiidynavox.com/devices/eye-gaze-devices/pceye-mini-access-windows-control/

Tobii Pro Glasses 2 Product Description. (2019). Retrieved May 28, 2019, from https://www.tobiipro.com/siteassets/tobii-pro/product-descriptions/tobii-pro-glasses-2-product-description.pdf/?v=1.95

Wilson, G. F., Fullenkamp, P., & Davis, I. (1994). Evoked potential, cardiac, blink, and respiration measures of pilot workload in air-to-ground missions. *Aviation, Space, and Environmental Medicine*, *65*(2), 100–105.

Zhai, S., Morimoto, C., & Ihde, S. (1999). Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 246–253). ACM.