Artificial Intelligence for Helicopter Safety: Head-Pose Estimation in the Cockpit

Eric Feuerstein
Graduate Student
Rowan University
Glassboro, NJ, US

Dr. Ghulam Rasool
Assistant Professor
Rowan University
Glassboro, NJ, US

Dr. Nidhal Bouaynaya
Associate Dean for Research and Graduate Studies
Rowan University
Glassboro, NJ, US

Dr. Ravi Ramachandran
Professor
Rowan University
Glassboro, NJ, US

Charles Johnson
Research Engineer, Flight Test Engineer, and Program Lead
Federal Aviation Administration
Egg Harbor Township, NJ, US

ABSTRACT

This paper discusses a novel method for improving rotorcraft safety. A new algorithm is proposed for estimating the head position of helicopter pilots and copilots using onboard cockpit videos. Cockpit videos offer the ability for crash analysts or incident investigation to understand not only the aircraft state but also the pilot and copilot's actions in a potentially unsafe situation. In addition, head pose information can also be used to improve the overall scanning techniques used by pilots or to research which technologies can assist the pilots in focusing more attention out of cockpit, rather than down at the instrument panel. Two algorithms were created to provide possible solutions to the problem of head pose estimation: a hybrid computer vision algorithm that utilizes deep learning based detectors, and a purely deep learning algorithm. The purely deep learning algorithm was able to correctly classify 91.49% of copilot head positions in a real-world flight video.

INTRODUCTION

The main objective of this work is to create an accurate head position estimation algorithm that is able to output the head positions of helicopter pilots and copilots from a supplied onboard cockpit video. The Federal Aviation Administration (FAA) continues to promote and highlight the importance of participating in aviation Flight Data Monitoring (FDM) programs to improve flight safety and operational efficiency. In addition, recorder safety was one of the topics on the agency's Top 10 Most Wanted List of Safety Improvements in 2017-2018 (Ref. 1). The FAA, National Transportation Safety Board (NTSB), and the United States Helicopter Safety Team (USHST) as well as other industry partners are working together to implement a helicopter safety enhancement that promotes the use of flight data recorders (FDR) to reduce the fatal accident rate in rotorcraft operations.

Although there is a need to integrate more FDRs into the rotorcraft community, certain obstacles still exist. The initial cost of FDRs can range from $9,000-$50,000 which does not include the cost to utilize them as part of an overall FDM program (Ref. 2). These costs alone play a significant role in preventing FDM programs from being adopted by small operators. On top of that, these devices can require technical expertise and special reading devices or software. Due to those reasons, rotorcraft typically have a lower participation rate in FDM programs than other forms of aviation (i.e. commercial fixed-wing or Part 121 airline operations) (Ref. 3).

On the other hand, even small helicopter operators often have the financial means to purchase one or more off-the-shelf video cameras which can be mounted inside the cockpit. These cameras, when pointed at the instrument panel, offer an alternate method of collecting the same data as traditional FDRs by utilizing post-processing of cockpit videos. Onboard video data also offers several possibilities for improving rotorcraft safety such as flight replay and the ability to extract information from where the pilot and copilot were focusing their attention during critical phases of flight. There is also an increase in the crash survivability of the data being collected because video information can be stored remotely.

This area of research also considers the obstacles faced when pilots transition from flying in Visual Meteorological Conditions (VMC) to Instrument Meteorological Conditions (IMC). VMC refers to clear visual scenarios where the pilot is able to fly by looking directly out the window and using visual references, while IMC refers to cloudy or obscure flight conditions where the pilot must fly using only the information presented on the instrument panel in front of them. Flying in

Presented at the Vertical Flight Society's 76th Annual Forum & Technology Display, Virginia Beach, Virginia, October 6-8, 2020.

IMC sometimes results in loss of control due to phenomena such as spatial disorientation, and usually result in fatal accidents. These events can often be traced back to improper instrument flight rule (IFR) scanning techniques or attention tunneling where the pilot fixates on a particular instrument at the expense of other instruments offering needed information. The rate of this happening can be reduced however, with better pilot training and with the added use of technologies such as enhanced/synthetic vision and heads-up displays. Gathering information about where the pilots are looking during critical phases of flight can be crucial in improving overall rotorcraft safety through better training, and for researching which new techniques and technologies allow for them to focus more of their attention outside the cockpit, rather than down at the instrument panel.

Motivation

The motivation of this research is to create a low-cost method using a combination of computer vision and deep learning techniques to determine the head position of helicopter pilots and copilots given onboard cockpit videos. Even in cases where a helicopter is equipped with an FDR, an investigator may not know what the pilot was focusing on during the moments leading up to or during a crash or incident. Cockpit video offers the ability to understand not only the aircraft state but also the pilot and copilot's actions in a potentially unsafe situation. The goal is to automate post-flight video processing and provide safety analysts or accident investigators with data on where a pilot was focused during any particular moment for any given flight. Admittedly, without the proper governance, this type of information could be used inappropriately by rotorcraft operators. However, the policies regarding the use of this information, while an important topic in its own regard, are outside the scope of this paper.

Before the initial implementation of the head pose estimation algorithm, the problem of gaze estimation for pilots was considered. This estimation technique looks at the eyes of the test subject in the videos and is able to determine exactly where the subject is looking at any given time. Since the problem is to determine how often a pilot is looking in a certain direction, gaze estimation would provide a very accurate prediction of this information. However, it is commonly found that pilots wear sunglasses or tinted face shields during their flights, and for that reason the eyes of the pilots are very frequently occluded from the camera's point of view. In addition, standard eye tracking cameras and technologies tend to fail during helicopter flight due to the excess vibration caused by the vehicle. As a substitute for gaze estimation, head pose estimation was selected as the next best choice for solving the problem at hand. While the exact location that the pilot is looking will be unknown due to the sunglasses, the general direction of gaze can be estimated using a head pose estimation technique.

Head pose estimation is a well-researched computer vision topic and is a solved problem when it comes to dealing with clean, passport-type photos. However, the challenge of identifying the head position at extreme angles with added noise and excessive background information is still a topic of discussion in the computer vision community. In most test videos supplied by the FAA, the pilots were looking at extreme angles and wearing helmets, sunglasses, microphones and other equipment that obstructed the camera's view of their face. Both a hybrid computer vision algorithm and a purely deep learning algorithm were created to classify the head positions of the pilots despite these additional obstacles. The computer vision algorithm presented in this paper classifies the head positions of the pilots into four main classes: (0) straight out the window, (1) down at the instrument panel, (2) out the window to the side, and (3) none of the above. The deep learning algorithm is capable of classifying the head positions into one of nine classes, allowing for a more fine-tuned head pose estimation: (0) Down, (1) Down_Left, (2) Down_Right, (3) Left, (4) Right, (5) Straight, (6) Up, (7) Up_Left, and (8) Up_Right. This information will be used for incident/crash analysis, future vision systems research at the FAA, and for improving the overall safety of rotorcraft operations.

Previous Work

There are a number of head pose estimation algorithms that exist already, and many of them deal specifically with passport-type photos, meaning the head in the image is frontal-facing and has limited noise or occlusion (see Ref. 4 for a survey). These methods commonly use a combination of 2D or 3D image points, facial feature extraction, tracking, or geometric approaches to solve the problem of head pose estimation.

In Ref. 5, a fusion method to head pose estimation is proposed. This method uses a frame-independent decision tree based estimator and a person-specific template tracker. A Kalman filter combines the estimations from both the estimator and tracker in real time to get accurate results using 3D depth images as well as 2D RGB information. By utilizing the 3D information from the depth camera, as well as the camera's intrinsic parameters, it is possible to recreate the 3D scene of the test subject's head position and train a head pose estimator based on point clouds. Also, the Dali3DHP dataset, was generated which has both depth and RGB information corresponding to each image. This method requires a person-specific template tracker to be trained beforehand, as well as depth camera information which is not available in the cockpit videos supplied by the FAA.

A semi-automatic method for facial landmark annotation is discussed in Ref. 6. Multiple, well-known facial databases including MutiPIE, XM2VTS, AR, and FRGC v.2 are used to train Active Orientation Models (AOMs) which are a type of deformable model used for model-based face analysis. The semi-automation comes from automatically applying facial landmarks from a known subset to an unknown subset and then manually classifying the fittings as "good" or "bad". The results in the paper are said to be so accurate that they can be used as ground truth. It is important to note that 87% of the

600 test images discussed in this paper have subjects with a horizontal pose variation angle between ±15° and the most 'extreme' angle considered is ±30°. Also, 70% of the test images are non-occluded images. There are very few, if any, annotated databases that consider truly extreme angles greater than ±30°. While this method is a feasible solution to annotating large datasets of frontal faces, the case of head poses at extreme angles is not considered.

An elegant a robust way to determine head pose by training a multi-loss convolutional neural network to predict Euler angles directly from image intensities is discussed in Ref. 7. This method emphasizes the fact that it does not require facial landmarks in order to accurately predict head pose. The model presented uses a prebuilt network architecture as a base but includes a multi-loss output approach which calculates a separate loss for each Euler angle (pitch, yaw, and roll). This allows for more fine-grained tuning of head poses to be obtained. While, this method does predict head pose angles in the presence of noise, a notable decrease in performance occurs in the presence of excessive background noise in the image. Without an efficient way to perform face detection or to crop most background information out of the FAA video data, this specific deep learning approach for predicting head pose is not feasible for the problem space discussed in this paper.

DATASETS

Four datasets were used throughout the course of this research. These datasets were used both for evaluation and validation of certain aspects of the hybrid algorithm, and for training and evaluating the deep learning models.

Head Pose Image Database

The first dataset is a publicly available benchmark dataset that was used to validate the accuracy of the hybrid algorithm. This dataset consists of fifteen subjects with ninety-three images corresponding to each subject for a total of 2790 monocular face images (Ref 8).

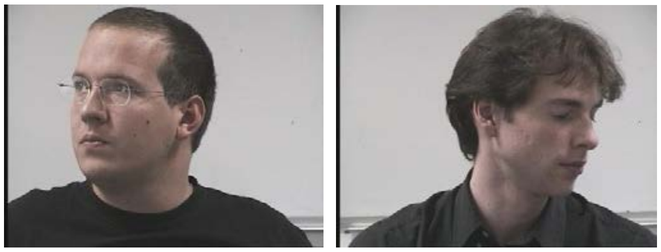

Figure 1. Sample images from the Head Pose Image Database.

Each image in the dataset has a corresponding pitch and yaw angle in degrees that serves as the ground truth for the image. The (vertical) pitch angles range from ±60° and the (horizontal) yaw angles range from ±90°.

Synthetic Dataset

The second dataset was created manually and consists of a series of synthetic test videos which display a subject wearing sunglasses and another with the subject not wearing sunglasses. This was done in order to simulate the conditions of the helicopter pilot test videos. This dataset was used heavily for the validation and experimentation of different aspects of the hybrid algorithm including face detection and facial landmark annotation.

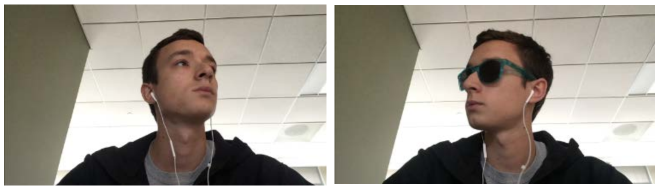

Figure 2. Sample images from the Synthetic Dataset.

FAA Flight Dataset

The third dataset was created using one of the real world flight videos provided by the FAA. This video consisted of 30976 copilot images and the first 10,000 images were manually labeled so a quantifiable accuracy could be calculated on both the hybrid and deep learning algorithms. Each image was given a class label between zero and eight: (0) Down, (1) Down_Left, (2) Down_Right, (3) Left, (4) Right, (5) Straight, (6) Up, (7) Up_Left, and (8) Up_Right. The naming convention was kept alphabetical and these labels are from the point of view of the camera, not the point of view of the copilot.

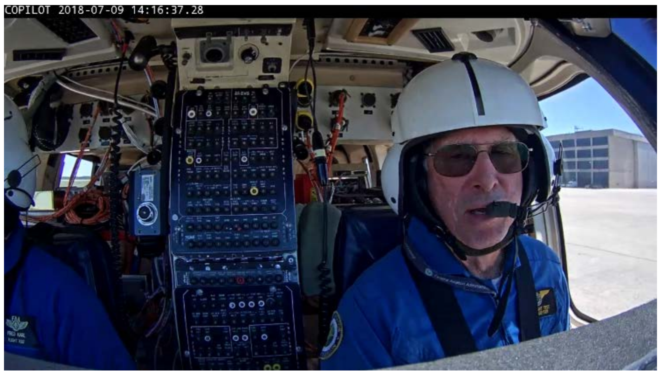

Figure 3. Sample images from the FAA Flight Dataset.

It is important to note that because the data was labeled manually, the ground truth is somewhat subjective. This stems from the fact that there is no way to set explicit boundaries to determine the exact time when a subject's head belongs to each class. Therefore, this dataset was created more to establish proof-of-concepts of the configured algorithms and to compare the results of both methods, rather than to quantify the true overall accuracy of the algorithm.

FAA Simulator Dataset

The final dataset was created after the completion of the hybrid head pose algorithm, and was used to train the deep learning models. The creation of this data was an extremely important step in this research and required a lot of fine tuning to ensure that a large amount of data could be collected quickly, while also making sure that the data simulated the real head positions of helicopter pilots during an actual flight. This data was created with the same nine classes from the FAA Flight Dataset.

Table 1. Class labels for the FAA Simulator Dataset.

Class Name	Label Number
Down	0
Down_Left	1
Down_Right	2
Left	3
Right	4
Straight	5
Up	6
Up_Left	7
Up_Right	8

Once the number of classes were defined, four test subjects of various heights, genders, and ethnicities were asked to sit in the pilot and copilot seats of the FAA's Sikorsky S76-D simulator located at the William J. Hughes Technical Center in Egg Harbor Township, NJ.

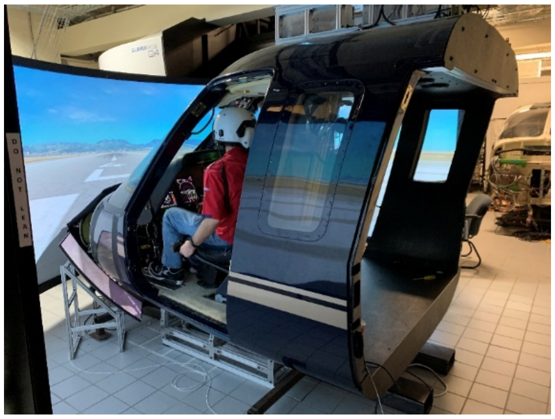

Figure 4. Sikorsky S76D simulator used for collecting training data.

An Axis S2016 NVR was used to record constant video of the subjects during each of the test runs. Each run consisted of the subject holding their head in the position of one of the nine classes for a total of one minute each. Each subject did not need to sit in both the pilot seat and copilot seat because the onboard cameras were positioned in such a way that the pilot video was almost an exact mirror of the copilot video when flipped on the y-axis.

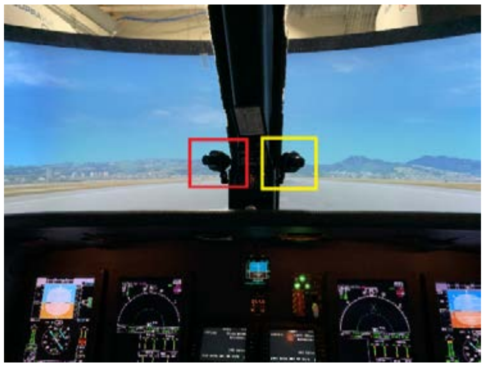

Figure 5a. Camera mounting positions for collecting data for the FAA Simulator Dataset. The box on the left was used to collect copilot data and the box on the right was used to collect pilot data.

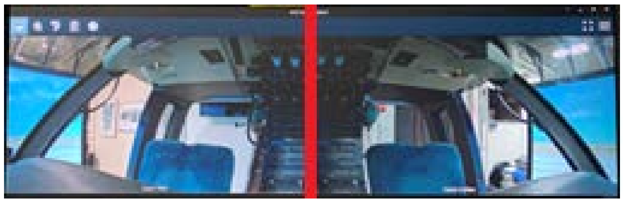

Figure 5b. The views from both cameras on the pilot side of the cockpit (left) and the copilot side (right).

In Figure 5a, the red box (left) shows the camera's mounting position for the copilot and the yellow box (right) shows the camera's mounting position for the pilot. Figure 5b shows the view from each of the cameras, separated by the bold red line in the middle. This configuration was done so that data augmentation could be used to create more training images without each subject having to actually switch sets. The pilot images were flipped over the y-axis and then used as copilot images, and vice versa. This allowed for double the amount of data to be collected in a shorter amount of time.

A total of six test runs were conducted for each subject to better simulate the various equipment that pilot's often wear during flights. The equipment that was worn during each of the test runs are shown in Table 2.

Table 2. Equipment worn during each test run while collecting data for the FAA Simulator Dataset.

Test Run	Equipment
1	Headset Only
2	Headset and Sunglasses
3	Helmet Only
4	Helmet and Sunglasses
5	Helmet with Clear Visor
6	Helmet with Dark Visor

An image from each of these test runs is displayed in Figure 6.

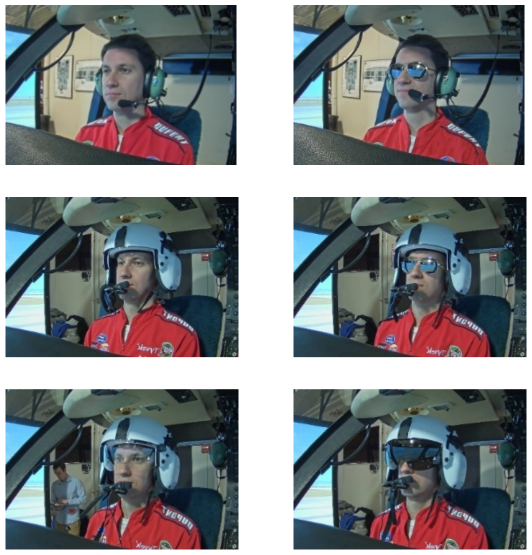

Figure 6. Sample images demonstrating the equipment worn by the test subjects for each test run when collecting data for the FAA Simulator Dataset.

A set of constraints were defined to limit the valid regions that a test subject could hold their head for each head pose class. Two diagrams (Figures 7a and 7b) show the valid regions for each vertical and horizontal movement of the test subjects.

In each view, the filled arrow shows the direction that the subjects head is facing and the curved line is the windshield of the helicopter. The horizontal movements of the head can be observed by looking at the top down view in Figure 7a. The dashed lines show the valid area that a test subject was allowed to look in each of the horizontal directions. The idea was to leave a buffer around the boundary between classes as much as possible, so that there would be a noticeable difference between the data that belonged to each of the nine classes.

The side view displayed in Figure 7b shows the vertical movements of the test subjects. After watching some real flight videos, it was observed that the difference between a pilot looking up, down, and straight is actually quite subtle, meaning they don't often look straight up or straight down at any point during the flight. For that reason, the test subjects were given specific points to look at to simulate real world flight data. When the subjects were looking down they were asked to look just at the bottom of the instrument panel and when they were looking up, they were asked to look directly at the intersection of the top of the windshield and the cockpit interior. These added constraints were vital to include when collecting the data because the deep learning networks needed to be trained on data that would simulate the real world test videos.

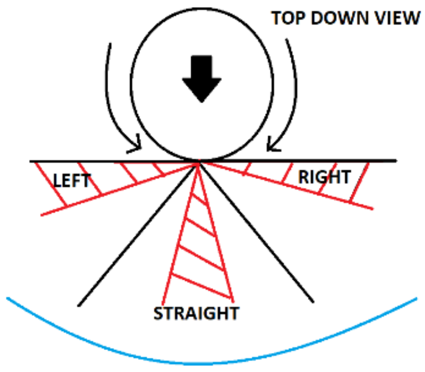

Figure 7a. Test subject constraints for horizontal head positions during data collection process. The filled in arrow shows the direction the subject is facing and the curved line is the helicopter windshield. The striped regions are the valid head position for a subject looking left, straight, and right.

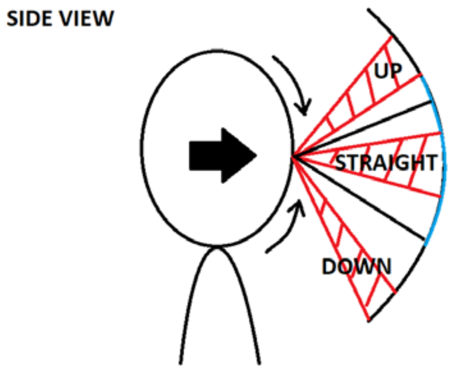

Figure 7b. Test subject constraints for vertical head positions during data collection process. The filled in arrow shows the direction the subject is facing and the curved line is the helicopter windshield. The striped regions are the valid head position for a subject looking up, straight, and down.

Once the test videos were created, the AxisFilePlayer software was used to crop the videos, convert them to images, and organize them into the proper directories. After the videos were organized accordingly, the data distribution was observed. The distribution for the pilot and copilot were exactly the same because the videos were cropped at exactly the same times. Table 3 displays the data distribution for both the pilot and copilot.

Table 3. Data distribution from collected simulator images.

Class Label	Total Number of Images
0	21853
1	22091
2	21990
3	22260
4	22181
5	21507
6	22079
7	21961
8	22028

This dataset is sufficiently large with approximately 21,000 images per class, and it is very evenly distributed. This data was used for the training and testing of each of the deep learning models presented in this paper.

HYBRID HEAD POSE ESTIMATION ALGORITHM

At the start of this research, a deep learning approach for making accurate predictions of the pilot's head position was considered. However, there were no labeled images of pilot's head positions available, and therefore a deep learning network could not be trained. Due to the lack of labeled ground truth data, a combination of classical computer vision techniques and deep learning based detectors, were originally selected as the best option to solve the problem of head pose estimation. The hybrid algorithm works with three main steps: (1) face detection, (2) facial landmark annotation, and (3) angle calculation for classification. The algorithm is performed on each frame of the supplied videos independently of one another and serves to classify images into one of four classes: (0) straight out the window, (1) down at the instrument panel, (2) out the window to the side, and (3) none of the above.

Face Detection

The first step of the algorithm is to locate the pilot's face in the image and draw a bounding box around it. To accomplish this, two face detectors were considered. The first detector uses a Histogram of Oriented Gradients (HoG) method to perform face detection and the second detector was trained using deep learning methods. The synthetic test videos were used to see which detector performed best on clean face images, and on noisier face images where the subject is wearing sunglasses. It was found that the deep learning detector accurately detected a face in all of the images from the clean synthetic video, and 98% of the images from the synthetic video with sunglasses. The HoG detector labelled much fewer frames than the deep learning based detector with an accuracy of 94% on the clean video and 70% on the video with sunglasses. Faces that were correctly detected by the deep learning based detector are displayed in Figure 8 below.

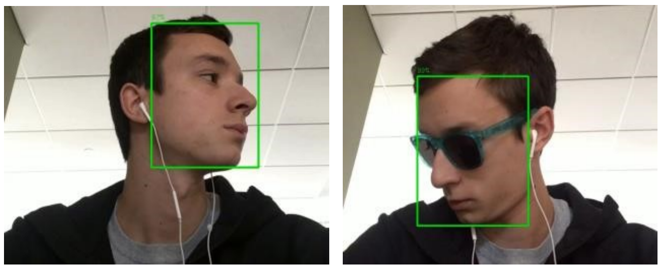

Figure 8. Correctly detected faces from the Synthetic Dataset using the deep learning face detector.

To further confirm that the deep learning detector was the best choice moving forward, the FAA Flight Dataset was considered. After providing each face detector with all images from the real world dataset, it was recorded that the HoG detector was able to detect 26.02% of the total faces whereas the deep learning detector detected 51.13% of the total faces. These accuracies are much lower than what was observed on the synthetic database due to the increase in background information, as well as the added noise of the pilot's helmet, microphone, and other equipment.

In an effort to increase the total number of faces detected, the frames were manually cropped to remove part of the excess background in the images. Due to the small cockpit space and limited mobility of pilots, the cropped area could be determined manually so that some background could be removed but the copilot never left the cropped region. After all frames were cropped, they were again supplied to the two face detectors. The HoG detector performed about the same, detecting 25.54% of the total faces and the deep learning detector was able to detect more faces at 75.20%. From these results, it is clear that the deep learning based detector is much more accurate and robust to noise than the HoG detector. From this point forward, only the deep learning based face detector was used.

A closer examination of the missed frames was conducted to better understand why faces were being missed by the detector. The most common case is when the copilot has turned their head so far to the side that their helmet is blocking their face from view. The second case occurs when occlusion causes the face to no longer resemble a face due to the added noise from the helmet, microphone, or other equipment. This can happen when the copilot is at an awkward angle or if their hands are blocking a portion of their face. To a human, it may be easy to identify a face in the presence of noise but to a computer that only sees an image as pixel values, it is much more difficult.

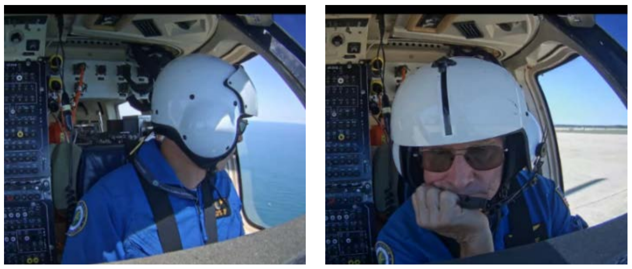

Figure 9. Faces that were not detected by the deep learning based face detector due to occlusion.

Partially occluded images, such as the image on the right side of Figure 9, are the most common cause for missed predictions when a face is actually present in the frame. On the other hand, it is important to remember that in a real world application, the detector will always miss video frames similar to the image on the left of Figure 9, where a helmet is blocking the face of the pilot or copilot.

An added benefit of the deep learning detector is that each detection comes with an accompanying confidence value. This value describes how sure the detector is that the detection is actually a face. This confidence metric is a valuable addition to the algorithm when a minimum confidence threshold is defined. This allows for weak detections or false positives to be eliminated if they do not meet the minimum threshold value.

After manually watching the real world flight video with the face detection overlay, it was observed that there was one major cause of false detection where a very small bounding box is placed incorrectly in the image. These false detections occur almost exclusively when the copilot is looking out the window to the side. Two examples of this are shown in Figure 10.

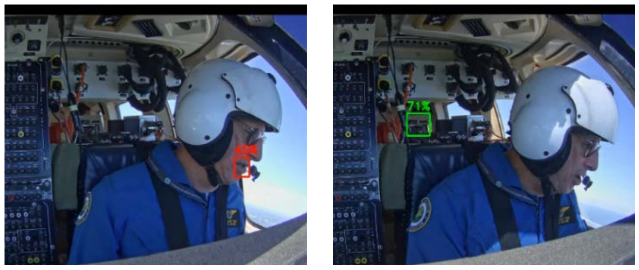

Figure 10. Two false detections by the deep learning based face detector where a small bounding box is placed in the background of the image.

These false detections can be easily removed by checking the diagonal distance of the predicted bounding box. Since the pilots are confined to a relatively small space, they cannot move enough that the size of their face's bounding box will change drastically from frame to frame. Therefore, the algorithm checks that the diagonal distance of the bounding box is greater than a manually defined value.

After the inspection of the false detections was conducted, various confidence thresholds were tested using the FAA Flight Dataset to find the optimal threshold value. The total number of faces detected as well as the total number of misdetections based on diagonal distance were recorded. Table 4 summarizes the results with confidence values ranging from 90% to 30% in increments of 10%.

Table 4. Confidence threshold test results.

Confidence Threshold	Total Frames with Face	Total False Positives	Total Frames (%)	False Positives (%)
90%	18455	0	59.57%	0%
80%	20587	8	66.46%	0.03%
70%	21729	34	70.14%	0.15%
60%	22516	102	72.68%	0.45%
50%	23294	164	75.20%	0.70%
40%	24035	256	77.59%	1.06%
30%	24888	391	80.34%	1.57%

As the confidence threshold decreases, the total number of frames with faces detected and the total number of false detections both increase. The optimal threshold value was selected to be 40% because it detects the largest amount of faces correctly while only having about a 1% misdetection rate. Figure 11 shows a few examples of frames where the face was detected accurately even at extreme angles and with some occlusion.

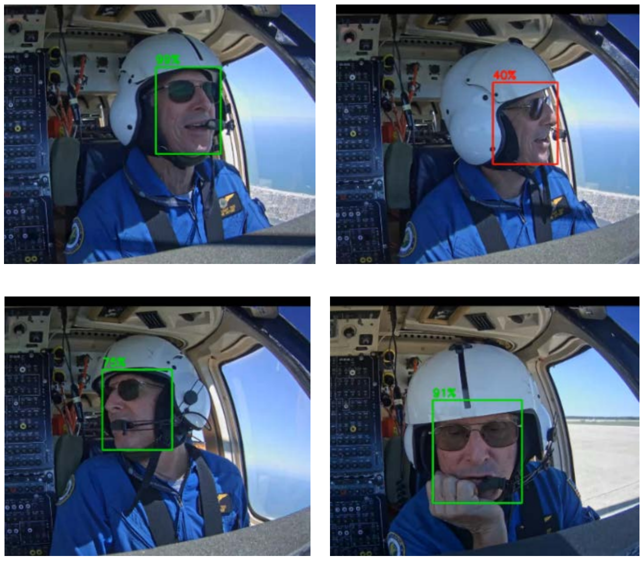

Figure 11. Correctly detected faces at extreme angles and with some occlusion.

This first step of face detection is the most important because the position of the bounding box is required for the next two steps of the algorithm. However, a method for estimating the head position when no face is detected is discussed in a future section.

Facial Landmark Annotation

The second step of the algorithm looks within the bounding box provided by the previous step and annotates important facial landmarks on the face. The facial landmark annotation tool (Ref. 6) that was selected for this step will always output 68 x-y coordinate pairs on the face which outline the jawline, eyes, eyebrows, nose, and mouth (Figure 12).

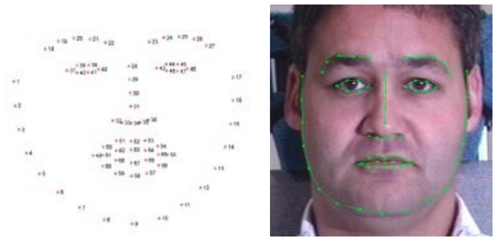

Figure 12. Layout of 68 facial landmark annotations.

Although there are 68 x-y pairs, the algorithm only considers six: the tip of the noise and chin, the two outside corners of the eyes, and the two outside corners of the mouth. These points were selected because they are sufficient in approximating the basic geometry of a face.

Figure 13 shows a few examples of correctly placed facial landmark annotations on real world copilot images.

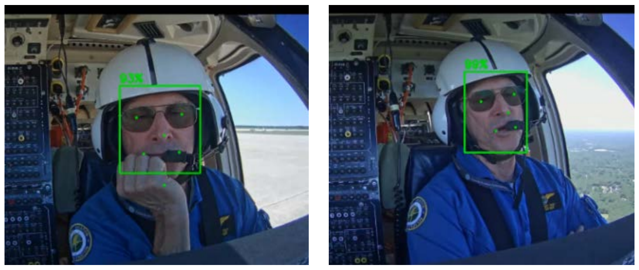

Figure 13. Properly placed facial landmark annotations.

The results are quite impressive because the facial landmarks are extremely robust to the added noise from the sunglasses and microphone, as well as occlusion from the copilot's hand. At frontal angles, these points are consistently and accurately placed on the copilot's face even with added noise. However, at more extreme angles, the facial landmarks are often skewed or wrong entirely as demonstrated in Figure 14.

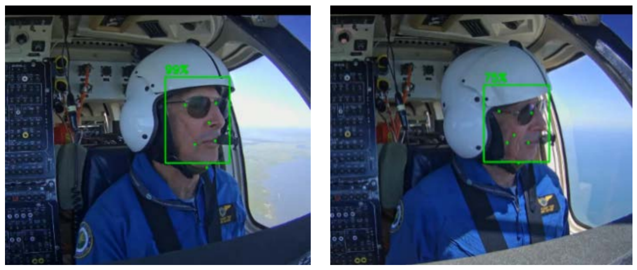

Figure 14. Misplaced facial landmark annotations.

Because this annotation tool was trained primarily on frontal faces, it is not a surprise that the facial landmarks are placed incorrectly when the copilot turns to extreme angles. However, it is found that in most cases, the annotations are only slightly off and still give a reasonable estimate of the copilot's head position. Because the goal of the algorithm is not labelling these points exactly, but rather, estimating the direction of the head, certain compensations can be made for poorly placed facial landmarks. This step of the algorithm was identified as one of the biggest limitations due to its inability to accurately place facial landmark annotations at extreme angles.

Angle Calculations and Classification

Once the positions of the facial landmarks are known, the next step is to use the pinhole camera model to obtain a rotation matrix and Euler angles that define the rotation of the pilot's head. In order to do this, a set of 3D reference points was manually defined from Ref. 9, such that they roughly model a head looking straight forward in 3D space. The reference model consists of the same six points that were annotated in the previous step: the tip of the nose and chin, the two outside corners of the eyes, and the two outside corners of the mouth. The 3D reference model and an example of 2D image points are displayed in Figure 15.

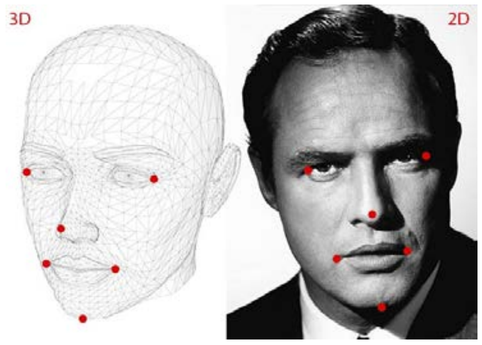

Figure 15. 3D reference model points (left) in comparison to 2D annotated points (right).

In order to obtain a rotation matrix using the 3D reference model and the 2D annotated points, the pinhole camera model is used. This model is commonly used in computer vision and it defines a mathematical relationship between the points in the 3D world coordinate system and their projection onto the 2D image plane of a pinhole camera. In simple terms, it is a method for approximating the mapping of points from a 3D scene, to points in a 2D image (Ref. 10).

Once the 3D reference points were defined, and the 2D image points are annotated from the previous step of the algorithm, a rotation matrix could be calculated. This rotation matrix contains all the information needed to calculate a pitch (α), yaw (β), and roll (γ) angle of the head. Pitch is defined as a vertical movement, yaw is defined as a horizontal movement, and roll is defined as a tilt of the head to either side.

Given a 3x3 rotation matrix, the formulas defined in Ref. 11 are used to calculate the angles of pitch, yaw, and roll. For the purpose of classification, only the pitch and yaw angles were considered for labeling any given head position in a video frame. After experimentation, it was concluded that the roll angle does not play a vital part in defining the direction a head is looking, and for that reason, it is omitted from the classification step of the algorithm. Each frame is applied a vertical or horizontal label depending on the pitch and yaw angles respectively. There a total of nine possible combinations of pitch and yaw labels and the threshold values for applying each label are shown in Table 5 below.

Table 5. Threshold values for applying pitch and yaw labels.

Label	Pitch (α)	Yaw (β)
Up	$\alpha > 10°$	-
Down	$\alpha < -10°$	-
Straight	$-10° < \alpha < 10°$	$-10° < \beta < 10°$
Left	-	$\beta < -10°$
Right	-	$\beta > 10°$

These values were determined by manually watching the real world copilot video and deciding when the copilot was considered to be looking in each direction. The threshold values will need to be manually adjusted depending on the conditions of the test video.

The combination of pitch and yaw label are used to classify the head position of the pilot into one of four classes for the hybrid algorithm: (0) Straight of the window, (1) Down at the instrument panel, (2) Out the window to the side, and (3) None of the above. The nine classes from the FAA Flight Dataset were organized into these four classes as given in Table 6.

Table 6. Combination of labels from the FAA Simulator Dataset that were used to classify head positions into the main classes of interest for the hybrid algorithm.

Class	Label Combinations (Pitch/Yaw)
0: Straight out the window	(3) Left (5) Straight
1: Down at the instrument panel	(0) Down (1) Down_Left
2: Out the window to the side	(2) Down_Right (4) Right (8) Up_Right
3: None of the above	(6) Up (7) Up_Left

The final output of the algorithm is an output video which has the class printed in the bottom left corner of the frame and a .csv file that contains the calculated head position of each frame. The algorithm has the option to display the face detection bounding boxes, facial landmarks, and angles if desired but will always output the class labels regardless of these other displays. In addition to the output video and output .csv file, the algorithm will also output the total number of frames that were labeled into each of the four classes.

Hybrid Compensation Method

As stated previously, the algorithm cannot estimate head pose angles if it doesn't first detect a face in the image. This limits the amount of correct predictions primarily in class 2: out the window to the side. In a real world scenario the copilot's helmet is almost always blocking their face from view when they are looking out the window, and therefore a face will not be detected.

After manually looking at the frames that had no face detected, it was observed that most of them were when the subject was looking out the window. Since the majority of missed faces do happen when the subject is looking out the window to the side, an assumption was made that if a face is not detected, the subject is most likely looking in that direction. The frame can then be labeled as belonging to class 2 even though a face is not present.

In order to prevent this assumption from also labelling cases of occlusion, each frame where no face is detected will check the previous frame's bounding box location to ensure that it is close to the window side of the cockpit. If the frame prior to no face being detected is close enough to the window, it is assumed that the subject continued to turn their head farther to the side to look out the window. The point that is considered "close enough" must be defined manually and will be dependent on the position of the camera in the cockpit.

DEEP LEARNING ALGORITHM

The hybrid head pose estimation algorithm works well for frontal facing poses and certain compensations were made to increase the accuracy at more extreme angles. However, this method does rely on a certain number of assumptions and approximations and will require a decent amount of calibration each time the conditions change. For that reason, it was decided that a ground truth dataset would be created and a purely deep learning approach would be used. This model would require less calibration in different scenarios and would be able to accurately predict head positions regardless of whether a face is present in the frame.

The goal of this deep learning algorithm is to create a fully automated algorithm, where the only input is the test video. The algorithm therefore should be able to tell automatically if the video is of a pilot or copilot, and be able to output the appropriate head pose class.

Dataset Organization

The process of collecting and labelling the data need to train a deep learning network has already been discussed but it is important to discuss the data distribution and data splits.

The entire labelled dataset is generally split into a training set, validation set, and test set. The training set is always the largest and can range anywhere from 60-98% of the total amount of labelled data. This data is used to help the network learn the optimal weights and biases needed to map the inputs to the outputs (Ref. 12). The validation and test sets are split evenly, based on the remaining percentage of labeled data that was not used in the training set. The validation set is passed through the network at the end of each forward pass during the training process to help measure how well the network is generalizing to an unknown dataset. The test set is used to evaluate the network's overall performance after training is complete. The network does not actually learn and update its parameters based on the validation and test sets, but these sets do provide important information on how the network will perform in real world scenarios after training.

It is important that the data be distributed evenly as well. In a multi-class problem, the number of labelled examples should be similar across all classes. The validation and test sets must also be of the same distribution as the training set. Without the proper organization of the labelled data, any network will be unable to produce accurate results.

From the total data collected, there were approximately 21,000 images in each of the nine classes, totaling 189,000 for the pilot and the same 189,000 images flipped over the y-axis for the copilot. A total of eight datasets were created by splitting the 189,000 images into four groups for the pilot and four groups for the copilot. A ninth dataset was created to determine if a video was on the pilot side or the copilot side of the cockpit.

The first grouping of data was created as the baseline dataset and consisted of a combination of helmet and headset images (with sunglasses, visors, etc.). However, after closer inspection of the data, it was a possible concern that the headset images and the helmet images that belonged to the same class might confuse the network during training, specifically when the pilot is looking out the window. This concept is demonstrated in Figure 16 below.

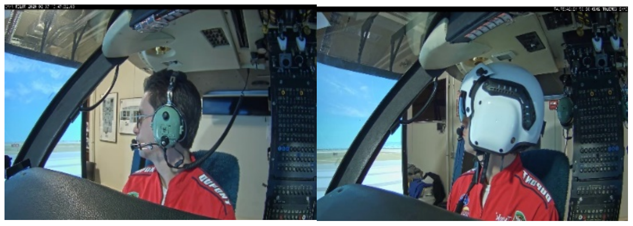

Figure 16. Variations between headset and helmet images within the same class.

For this reason, two more groups of data were created: one that contained nine classes but only images of the pilot/copilot with a headset on, and another that contained nine classes but only images of the pilot/copilot with a helmet on. The point of separating the data into these two groups was to increase the overall accuracy of the algorithm especially in the case where the pilot/copilot is looking out the window. Due to the limited amount of labeled data at the time of this research, these separate helmet/headset datasets have the potential to offer a more accurate prediction than the combined dataset. In addition, having one combined prediction and one headset/helmet prediction also adds some redundancy to the algorithm and can help analysts or interpreters to identify misclassifications if they see a significant difference between the two predictions.

Additionally, a two class dataset was created to identify the headgear that the pilot/copilot is wearing in the test videos. This dataset has two classes where the first class consists of headset images and the second class contains helmet images. The same images from all nine classes in the previous datasets were reused but reorganized into these two new classes. All the headset images were used totaling 63,000 images in class 0, and this number was matched with helmet images in class 1 to ensure an even data distribution. The images in this dataset were selected at random from all available helmet images belonging to all nine classes.

Finally, a two class dataset was created to identify which side of the cockpit the video was taking place. This dataset was created to further automate the video processing step of the algorithm.

Each dataset was split into a training/validation/test split using a distribution of 90/5/5 respectively. A summary of the datasets and approximate distributions are provided in Table 7. The numbers listed are for the pilot datasets but the numbers will be identical for the copilot datasets.

Table 7. Summary of datasets used in the deep learning algorithm.

Dataset Name (Classes)	Training Images per Class	Test/Validation Images per Class
PCombined (9)	19,800	1,100
PHeadset (9)	6,660	370
PHelmet (9)	13,140	730
PClassifier (2)	59,742	3,319
HelicopterSide (2)	30,000	1,500

Model Selection and Hyperparameter Tuning

The challenge of any deep learning approach is that the hyperparameter values such as learning rate, dropout rate, and pooling that will produce a viable solution are almost always unknown. On top of this, it is also difficult to know what network architecture will work best given the data available. If the data is very complex, the network needs to be deep enough to learn complex features, but if the data is simpler, a deep network may cause overfitting. In order to tune the hyperparameters and to figure out which network architecture will work best for the task of head pose estimation, multiple network architectures were trained with different combinations of hyperparameters in order to explore a variety of possible solutions.

The coding framework that was used for the training and evaluating process was Keras. This framework provides users with Keras Applications which allow for the easy implementation of preconfigured network architectures. There are a wide variety of network architectures that are available in Keras Applications and seven of them were considered in the initial testing phase of this research. The architectures include: ResNet50, VGG16, VGG19, InceptionV3, Xception, InceptionResNetV2, and DenseNet121. An output layer was added to each of these networks to ensure that the output of the network had the correct number of classes for the dataset being used.

Alongside different network architectures, different combinations of hyperparameters were used. Table 8 below shows the possible values of each hyperparameter. The pooling in this table refers to the pooling applied to the last layer only.

Table 8. Selected hyperparameter values that were considered for model training.

Learning Rate	Dropout Rate	Pooling
0.0009	0.5	None
0.005	0.25	Average
0.001	0.1	
0.01	0	

The first dataset that was considered for training was the PHelmet_9Class dataset. This nine class network consisted of pilot helmet images only and was selected as the dataset for initial testing for a few reasons. First, it was smaller than the full dataset, allowing for multiple models to be trained quickly, and second, the images in this dataset are generalizable to the images in the other eight datasets. For each of the seven network architectures listed above, a total of eight models were trained with different hyperparameter combinations. The initial results from these 56 trained models were used to narrow down the search for well-performing combinations of architecture and hyperparameters.

In order to evaluate each of the models, the model weights and network architectures were saved after each training session. The architectures and weights were then used to get a head pose prediction for each image in the test set. The networks did not see these test images at any time during the training process so the testing accuracies represent the generalizability of the model on unknown data. The predictions for each image were recorded and aligned in a confusion matrix so the accuracy of each class could be observed.

A confusion matrix is a representation of how well a deep learning model is performing. An example is shown in Figure 17.

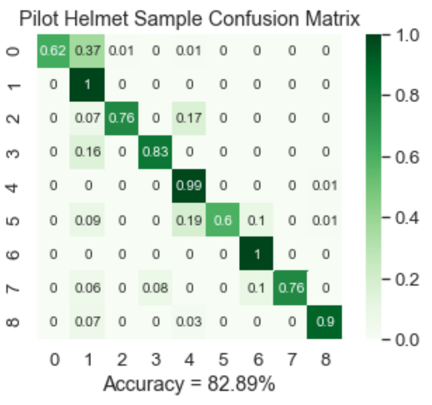

Figure 17. Sample confusion matrix.

Each block of a confusion matrix represents how many test images from a specific class were classified into each of the nine classes. For example, the top row of the confusion matrix in Figure 17 represents all the test images from class 0, and the columns represent the output predictions of the model. The blocks in the first row show that 62% of the test images belonging to class 0, were correctly classified as class 0. However, 37% of the test images from class 0 were classified as class 1, and 1% were also classified as class 2 and 4. Looking at the second row from the top, it shows that 100% of the test images from class 1 were correctly classified as class 1.

A model that is 100% accurate on the test set should ideally have a confusion matrix with a diagonal of 1's going from the top left corner to the bottom right corner. The total accuracy of the model is calculated by taking the average of all the diagonal blocks in the confusion matrix, because these blocks represent correct predictions in the correct class.

The confusion matrices for all 56 models trained on the PHelmet_9Class dataset were observed in order to gain more information on which architectures performed well on the data and which combination of hyperparameters outperformed others. The architecture, learning rate, dropout rate, and pooling for the best performing models were recorded along with the overall accuracy of each model on the test set.

From the initial results it was clear that the networks performed best with a relatively small learning rate and at least some percentage of dropout. For that reason, the dropout rate of 0 and the learning rate of 0.01 were removed from testing in the future. The most promising results were obtained in most cases using a learning rate of 0.0009 and a dropout rate of 0.5, with average pooling and no pooling both working in some cases. It was also observed that the VGG16 and VGG19 architectures produced poor results regardless of the hyperparameters.

For the remaining eight datasets, the Xception, InceptionV3, and ResNet50 architectures were considered. These architectures were selected because they each have their own unique aspects to them whether it is using residual blocks (Ref. 13), inception modules (Ref. 14), or a combination of both (Ref. 15). By narrowing down the architectures and hyperparameter combinations moving forward, the number of models trained for each dataset was greatly reduced from 56 models to 18 models.

Once the number of networks was reduced, the 18 different models were trained on each of the four copilot datasets. The copilot datasets were considered next in order to verify that the hyperparameters and architectures that produced good results on the pilot helmet dataset also produced good results on the copilot datasets. The results and overall accuracies of the 18 models were recorded for each of the copilot datasets and the model with the best overall accuracy was highlighted.

After the copilot models were trained, the remaining pilot models and the helicopter side model were trained. Rather than training 18 networks per pilot dataset, only one or two networks were trained using the combination of architecture and hyperparameters that produced the best results on the corresponding copilot dataset. This assumption was made because the pilot images are similar to the copilot images except that they are flipped over the y-axis. The best results from the copilot classifier were used to train the helicopter side model.

Final Algorithm Structure

Once there was a working model for each of the nine datasets, the final structure of the algorithm was designed (Figure 18). Each frame of the input video passes through a total of four networks, and the predictions of each frame are performed independently of one another.

$$\begin{tikzcd}
 & & \text{HELICOPTER SIDE} \arrow[dll] \arrow[dl] \arrow[dr] \arrow[drr] & & \\
\text{P\_COMBINED} & \text{P\_CLASSIFIER} \arrow[dl] \arrow[d] & & \text{CP\_CLASSIFIER} \arrow[d] \arrow[dr] & \text{CP\_COMBINED} \\
\text{P\_HEADSET} & \text{P\_HELMET} & & \text{CP\_HEADSET} & \text{CP\_HELMET}
\end{tikzcd}$$

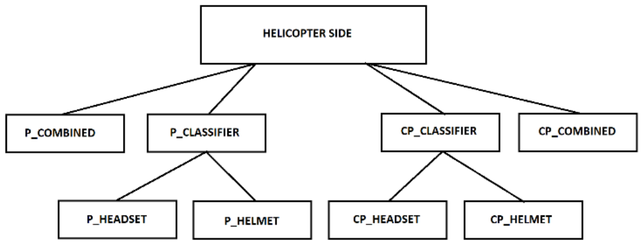

Figure 18. Deep learning algorithm structure.

The input image will first pass through the helicopter side model to determine if the video is of a pilot or a copilot. Depending on that prediction, the image will then be given to both the combined model and the headgear classifier model. The output from the headgear classifier will then determine if the image will be given to the headset model or helmet model. Each image will have an accompanying helicopter side prediction, combined prediction, classifier prediction, and either a headset or helmet prediction.

The final output of the deep learning algorithm is an output video and an accompanying .csv file. All four predictions for each frame will be printed on the output video and these four predictions will be saved to a .csv file with its appropriate time stamp and frame number. A summary of the total number of frames classified into each class is also included at the bottom of the .csv file.

Up to this point all classes were labeled from the point of view of the camera. However, it is known that the actual direction of the head pose should be from the point of view of the pilot/copilot. For that reason, all labeled data for the training and testing of the networks will remain from the camera's point of view, but the display on the output video and the predictions in the .csv file are changed to be from the pilot/copilot's point of view.

Generalizing to a Real World Dataset

The models discussed in the previous section were trained on simulator data only and no real flight data was included. That being said, these simulator models did not perform well on the images from the FAA Flight Dataset because of the difference between the two sets of images. This difference is displayed in Figure 19.

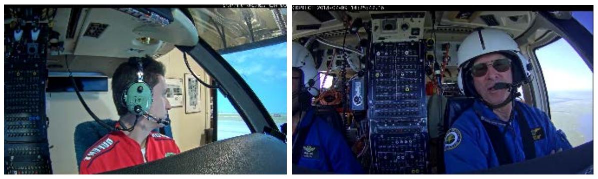

Figure 19. Simulator data (left) compared to real flight data (right).

Although the camera angle may not look much different to the human eye, remember that a computer only sees an image as pixel values. This means that any change of background, average pixel color, or camera angle can drastically affect the ability of the model to provide accurate predictions.

Since this algorithm will be used for real flight video data and not just simulator data, it is important that the models generalize to new cockpits, camera angles, or testing conditions. In order to generalize to more situations, more variations in the training data are required. The simulator data must be used at this stage of the research because labeled head pose data is limited, however future training data variations should include slight camera angle adjustments, different cockpit interiors, different pilots/copilots, different headgear/equipment, and different flight scenarios such as daytime or nighttime flights.

To demonstrate the process of generalizing the models to a set of real world data, images from the real world copilot video were labelled manually and included in the training data along with the simulator images. From the thirty minute test video, the first ten minutes were set aside for final testing, and the images from the second twenty minutes were manually labeled into each of the nine classes. The total data distribution from this manual labeling is shown in Table 9.

Table 9. Data distribution from collected simulator images.

Class Label	Total Number of Images
0	761
1	186
2	919
3	2339
4	4141
5	6678
6	1175
7	1328
8	380

A total of 17,907 images were collected across all nine classes. While there is a clear imbalance in the new images shown in Table 9, this imbalance became less obvious when these images are added and shuffled in with the current copilot datasets that contained only simulator data. The total data distribution with both the combined simulator data and the additional 17,907 images from the copilot test video is displayed in Table 10.

Table 10. Distribution of total copilot images after simulator data and real world data were combined.

Class Label	Total Number of Images
0	22614
1	22277
2	22909
3	24599
4	26322
5	28185
6	23254
7	23289
8	22408

Once these new images were added to the training set, four of the nine models were retrained with the real world copilot images included. By adding only a few more examples to each class from this new dataset, the networks were able to perform well on real flight data without sacrificing their accuracies on the simulator data.

RESULTS

This section will contain a full presentation of the results of both the hybrid computer vision algorithm and the deep learning models that were trained throughout this research. The performance of the hybrid algorithm on a benchmark dataset and a real world dataset will be explored. The effects of the hybrid compensation method for classifying images where no face is detected will be discussed as well. The training and validation accuracies of each deep learning model are presented as well as the best confusion matrix that was obtained for each model.

Hybrid Head Pose Estimation Algorithm Results

The first experiment that was conducted for testing the accuracy of the hybrid head pose estimation algorithm was performed on the Head Pose Image Dataset. As stated previously, this benchmark dataset is made up of 2790 images, each with a corresponding pitch and yaw label to define the position of the head in the image. The absolute error between the true pitch and yaw angle and the algorithm's calculated pitch and yaw angle was calculated for all 2790 images. Once the error was calculated for every image, the average absolute error for each individual pitch and yaw angle was obtained. The average absolute error is displayed in Figures 20 and 21.

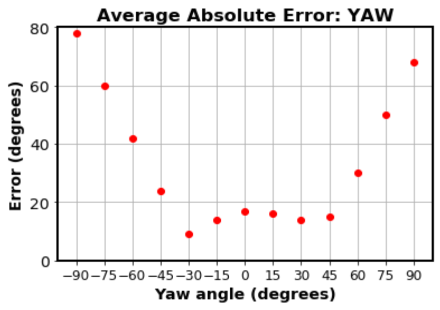

Figure 20. Average absolute error calculated by the hybrid algorithm on the Head Pose Image Dataset for all possible yaw angles.

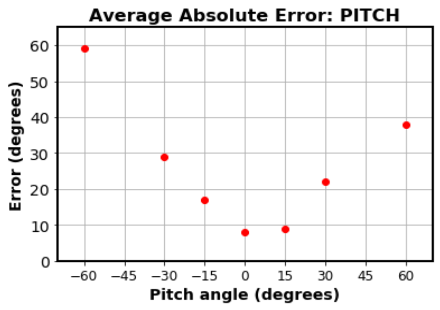

Figure 21. Average absolute error calculated by the hybrid algorithm on the Head Pose Image Dataset for all possible pitch angles.

From these two figures it is clear to see that as the head angle becomes more extreme, the absolute error increases very quickly. The algorithm performs well in calculating yaw angles in the range of ±30° and pitch angles in the range of ±15°. This is not an unexpected result because of the fact that the facial landmark annotation tool was only trained on faces in these ranges. Figures 20 and 21 show that the tool does provide accurate annotations within that range, and verifies that the algorithm works very accurately for frontal facing poses within a certain range.

After collecting a quantifiable accuracy of the angles calculated by the hybrid algorithm, the next experiment was performed on the FAA Flight Dataset. The first 10,000 frames of this flight video were manually labeled into nine classes so the output classifications from the algorithm could be compared to ground truth values. Table 6 in the Angle Calculation and Classification section displays how the nine classes from this dataset were divided into the four classes of interest for the hybrid algorithm. Table 11 shows the total number of frames belonging to each class.

Table 11. Total number of frames belonging to each class of the FAA Flight Dataset.

Class	Total Frames
0: Straight out the window	5629
1: Down at the instrument panel	627
2: Out the window to the side	1859
3: None of the above	1885

From Table 11, it is apparent that the majority of the time during flight, the copilot is looking straight out the window. That being said, the algorithm should be able to perform well on this class specifically. Figure 22 shows a correctly classified image from the first three classes.

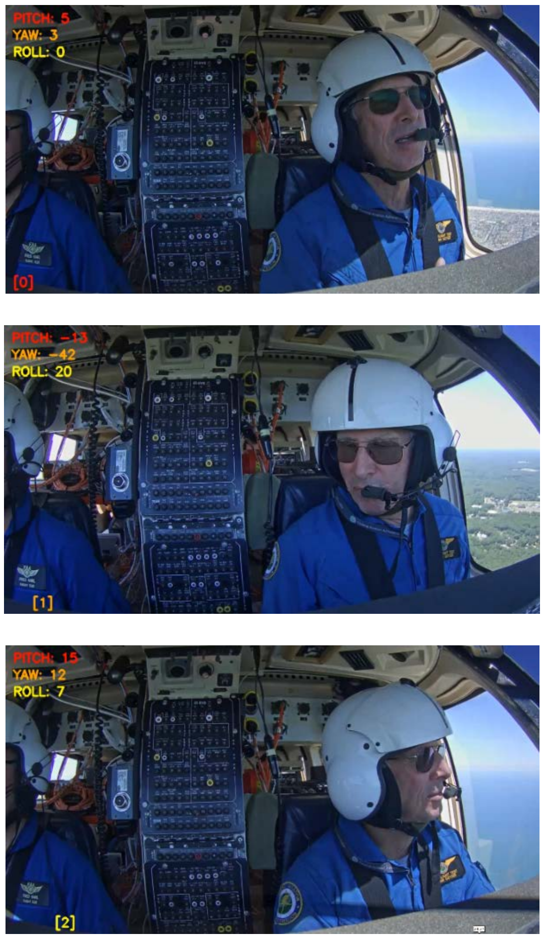

Figure 22. Frames that were correctly classified by the hybrid algorithm.

After supplying all 10,000 test images to the hybrid algorithm and comparing its classifications with the ground truth, the overall accuracy was calculated to be 46.01%. While this accuracy does seem quite low, it is important to look at the accuracy of each class individually by observing the confusion matrix in Figure 23.

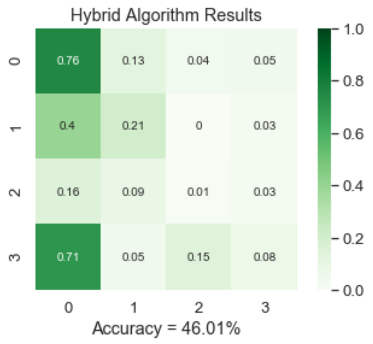

Figure 23. Confusion matrix for standard hybrid algorithm when evaluated on the FAA Flight Dataset.

The algorithm performs best in classifying frames where the copilot is looking straight out the window, correctly classifying 76% of those images. Of the 10,000 labeled test frames, about 56% of them belong to class 0. Therefore, it is a good sign that the algorithm performs well in that class specifically. However, the correct classifications for class 1 and class 2 are much lower than would be desired. The confusion matrix also clearly shows that when the algorithm classifies a frame incorrectly, it is frequently classifying the frame as straight out the window. This should not be a surprise, again because the facial landmark annotation tool was trained on frontal faces only.

Looking more closely at the incorrectly classified images, a few conclusions can be made. First, the algorithm struggles to classify images into class 1, down at the instrument panel, because the difference between the copilot looking straight and the copilot looking down is quite subtle. Therefore, the facial landmarks that are annotated onto the face need to be very accurate in order to detect this small change from class 0 to class 1. Due to the added noise from the copilot's sunglasses, microphone, and other equipment, the facial landmarks are not as accurate as they would be on a clean image with no noise. Therefore, the inaccuracy of the facial landmark annotations due to this added noise was concluded to be the major reason for the lack of correctly classified frames in class 1.

On the other hand, the algorithm does not classify frames into class 2, out the window to the side, because of the limitations of the face detector. It was found that from the 1859 frames belonging to class 2, the face detector was unable to detect a face in 70% of them. As stated previously, a face must be detected in the frame in order to get a head pose classification. This is a huge limitation for the overall accuracy of the algorithm. However, a method for classifying head positions where no face is detected was included in a second version of the algorithm. After implementing this new method and calculating the overall accuracy again, it was found that the new accuracy of the algorithm with the added compensation method was 55.26%, an increase in the overall accuracy of about 9%.

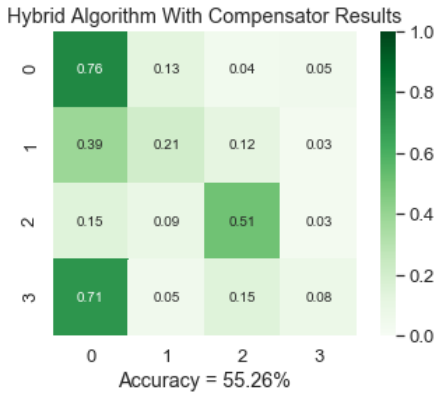

Figure 24. Confusion matrix for the hybrid algorithm with the added compensation method when evaluated on the FAA Flight Dataset.

Looking at the confusion matrix in Figure 24, it is clear that rows 0, 1, and 3 have not changed much from the confusion matrix in Figure 23. However, the main difference can be viewed in that there was an increase in the accuracy of class 2 by 50%. This improvement shows the validity of the assumptions made for the compensation method. Of the 70% of frames that had no face detected, almost 1000 of these frames were now correctly classified as belonging to class 2 even though no face was detected. One image that was previously labeled as having no face detected but is now labeled correctly to class 2 is displayed in Figure 25.

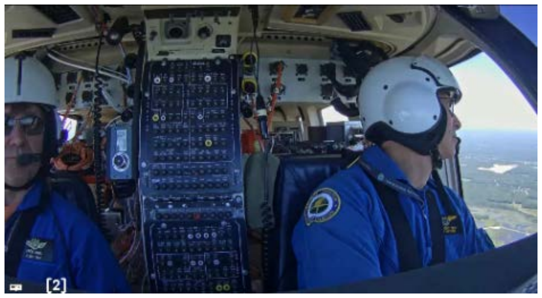

Figure 25. Correctly classified frame from the FAA Flight Dataset where no face was detected.

From these results it can be said that the hybrid algorithm proposed can provide an accurate estimation of helicopter pilot head pose in certain scenarios. Since the pilots are looking straight ahead most of the time (56% in the first ten minutes of this test video) the algorithm is a valid solution for predicting these head poses. However, there are some obvious limitations when it comes to detecting small changes between classes and when the head position is at an extreme angle. Certain methods have been implemented to overcome these limitations, but there is more to be desired in terms of accuracy in the extreme angle case. For that reason, these initial results were used as motivation to create a true ground truth dataset so that a more accurate deep learning model could be trained. This deep learning algorithm would be able to predict head poses at both frontal angles and at extreme angles, regardless of whether or not a face is present in the frame.

Deep Learning Simulator Model Results

A total of 18 models were trained on each of the four copilot datasets for a total of 72 copilot models. The network architectures considered in these tests were the ResNet50, InceptionV3, and Xception architectures. A total of six models were trained for each architecture using the following combinations of hyperparameters shown in Table 12.

Table 12. Hyperparameter combinations for each of the six models trained for each of the three selected network architectures.

Model	Learning Rate	Dropout Rate	Pooling
1	0.0009	0.5	Average
2	0.0009	0.5	None
3	0.005	0.25	Average
4	0.005	0.25	None
5	0.001	0.1	Average
6	0.001	0.1	None

The copilot datasets were trained with simulator images only, and the hyperparameters that produced the best results for each of the four datasets are displayed in Table 13.

Table 13. Summary of hyperparameters that produced the best results for the copilot models.

Network (Classes)	Architecture	Learning Rate	Dropout Rate	Pooling
CPHelmet (9)	Xception	0.0009	0.5	Average
CPHeadset (9)	Xception	0.0009	0.5	Average
CPCombined (9)	Xception	0.0009	0.5	Average
CPClassifier (2)	ResNet50	0.0009	0.5	None

From this table it is clear to see that the best learning rate and dropout rate in all cases was 0.0009 and 0.5 respectively. Average pooling seemed to perform well on the nine class datasets, where no pooling in the last layer resulted in better performance on the two class headgear classifier. The Xception architecture also outperformed the other architectures on the nine class datasets, and the ResNet50 architecture performed best on the data with only two classes. The training accuracy, validation accuracy, and test accuracy for each of the best models are shown in Table 14.

Table 14. Summary of copilot model accuracies.

Network (Classes)	Training Accuracy	Validation Accuracy	Test Accuracy
CPHelmet (9)	99.84%	99.96%	98.67%
CPHeadset (9)	99.71%	99.93%	99.89%
CPCombined (9)	99.91%	100%	99.11%
CPClassifier (2)	99.98%	99.98%	100%

The training and validation accuracies are expected to be high because that confirms that the networks are learning the important features and information from the training images. The test accuracy was calculated using a test set of images that was not shown to the network at any time during the training process. Observing a small difference between the training, validation, and test accuracies validates that the model is not overfitting to the training data and is remaining generalizable to data that it has not seen before. The confusion matrices for these four models are shown in Figure 26.

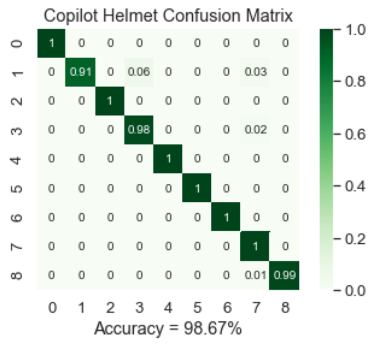

Figure 26a. Copilot helmet confusion matrix trained on simulator data only. This model was trained using the Xception network architecture, a learning rate of 0.0009, a dropout rate of 50%, and average pooling in the last layer.

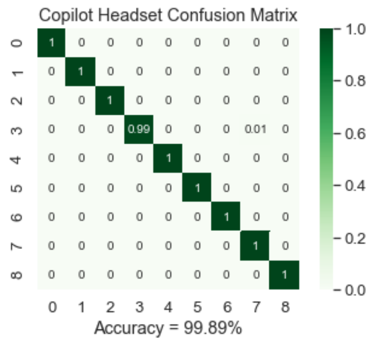

Figure 26b. Copilot headset confusion matrix trained on simulator data only. This model was trained using the Xception network architecture, a learning rate of 0.0009, a dropout rate of 50%, and average pooling in the last layer.

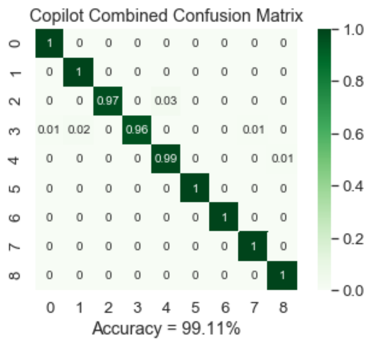

Figure 26c. Copilot combined confusion matrix trained on simulator data only. This model was trained using the Xception network architecture, a learning rate of 0.0009, a dropout rate of 50%, and average pooling in the last layer.

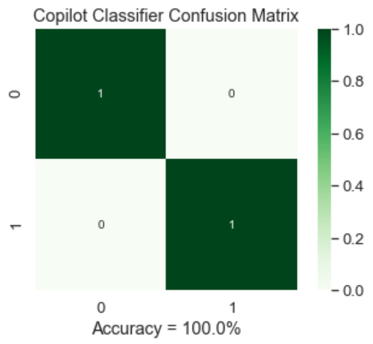

Figure 26d. Copilot headgear classifier confusion matrix trained on simulator data only. This model was trained using the ResNet50 network architecture, a learning rate of 0.0009, a dropout rate of 50%, and no pooling in the last layer.

Once the four copilot datasets had working models, the same combinations of hyperparameters from Table 14 were used to train a single model for each of the four pilot datasets. These hyperparameters work for both pilot and copilot images because these images are essentially the same just flipped over the y-axis. The accuracies for the pilot models are shown in Table 15.

Table 15. Summary of pilot model accuracies.

Network (Classes)	Training Accuracy	Validation Accuracy	Test Accuracy
PHelmet (9)	99.87%	100%	99.78%
PHeadset (9)	99.82%	100%	99.89%
PCombined (9)	99.95%	100%	99.33%
PClassifier (2)	99.87%	100%	99.50%

Again, there is a small difference between training accuracy and testing accuracy which shows that the network is not overfitting. The accuracies are very high again because the variations in the data are limited at this time. The confusion matrix for each model is displayed in Figure 27.

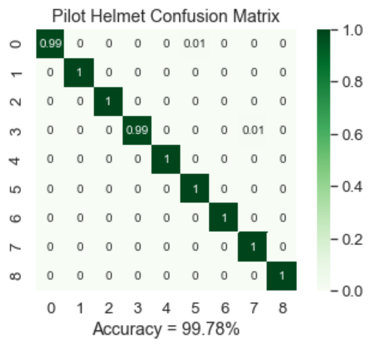

Figure 27a. Pilot helmet confusion matrix trained on simulator data only. This model was trained using the Xception network architecture, a learning rate of 0.0009, a dropout rate of 50%, and average pooling in the last layer.

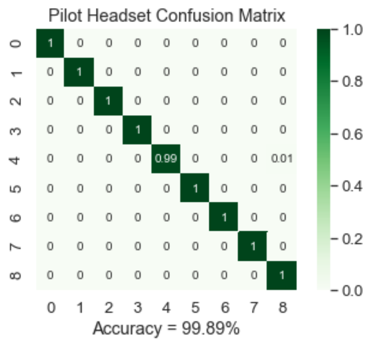

Figure 27b. Pilot headset confusion matrix trained on simulator data only. This model was trained using the Xception network architecture, a learning rate of 0.0009, a dropout rate of 50%, and average pooling in the last layer.

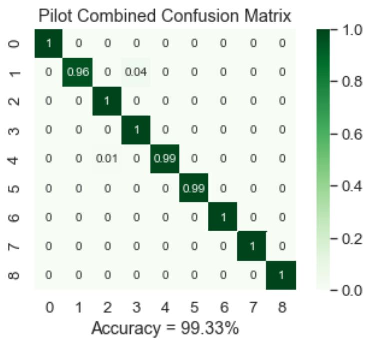

Figure 27c. Pilot combined confusion matrix trained on simulator data only. This model was trained using the Xception network architecture, a learning rate of 0.0009, a dropout rate of 50%, and average pooling in the last layer.

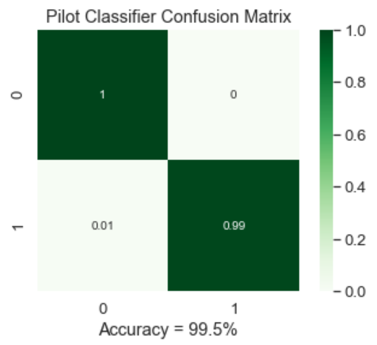

Figure 27d. Pilot headgear classifier confusion matrix trained on simulator data only. This model was trained using the ResNet50 network architecture, a learning rate of 0.0009, a dropout rate of 50%, and no pooling in the last layer.

The final network to analyze was the helicopter side network. This network was trained using the same combination of hyperparameters as the pilot and copilot head gear classifier models.

Table 16. Hyperparameter Summary of Best Results.

Network (Classes)	Training Accuracy	Validation Accuracy	Test Accuracy
HelicopterSide (2)	100%	100%	95.5%

Table 16 shows that the model had extremely high accuracies on both the training and validation data meaning that the features for determining which side of the cockpit the video is on are fairly simple to learn. The confusion matrix for this model displayed in Figure 28.

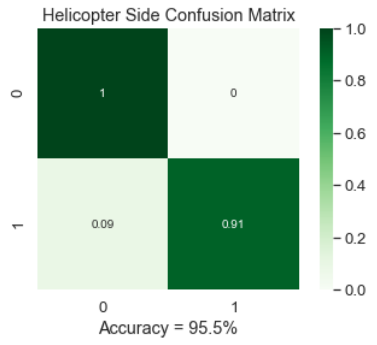

Figure 28. Helicopter side confusion matrix trained on simulator data only.

These initial results on the simulator data show that a working model with high accuracy was achieved for each of the nine models within the final algorithm structure. A few correctly classified images from the test sets are shown in Figure 29 and Figure 30. The text in the top left of the image is the prediction from the headgear classifier (white), under that is the headset/helmet prediction (green/blue respectively), and the beneath that is the combined prediction (red). The helicopter side prediction, point of view, and frame number are printed in yellow at the bottom of the images.

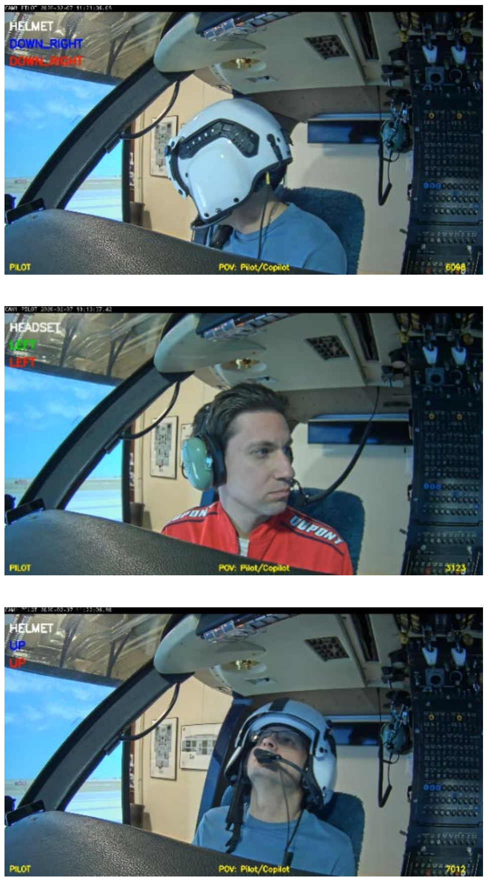

Figure 29. Pilot frames correctly classified by simulator models. The white text in the top left is the prediction of the headgear classifier. The green/blue text and red text are the nine class head pose predictions for the headset/helmet model and combined model respectively. The yellow text at the bottom displays the helicopter side prediction, prediction point of view, and frame number.

These images demonstrate the deep learning algorithm's ability to correctly predict the head pose of pilots and copilots regardless of whether a face is present in the frame. The algorithm also consistently predicts the correct helicopter side as well as the head gear worn by the pilot and copilot. Figure 29 and 30 also display examples of the combined predictions in red being consistent with the headset/helmet predictions in green/blue. When both head pose predictions are the same, it gives added confidence that the prediction is correct.

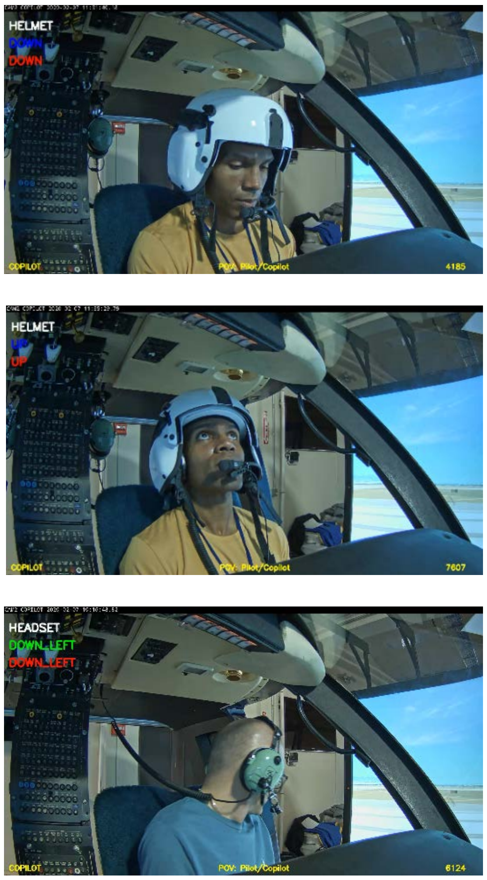

Figure 30. Copilot frames correctly classified by simulator models. The white text in the top left is the prediction of the headgear classifier. The green/blue text and red text are the nine class head pose predictions for the headset/helmet model and combined model respectively. The yellow text at the bottom displays the helicopter side prediction, prediction point of view, and frame number.

While these examples quite clearly demonstrate the algorithm's success, there are also certain conditions that can cause the algorithm to struggle with making accurate head pose predictions.

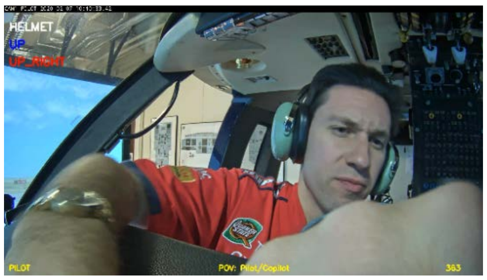

Figure 31a. Pilot frame incorrectly classified by simulator models due to occlusion.

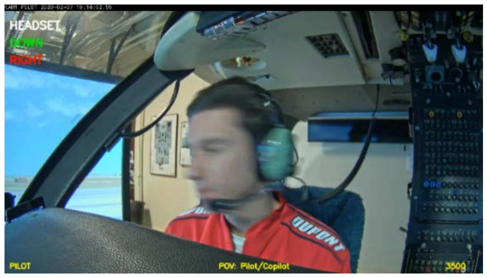

Figure 31b. Pilot frame incorrectly classified by simulator models due to image quality.

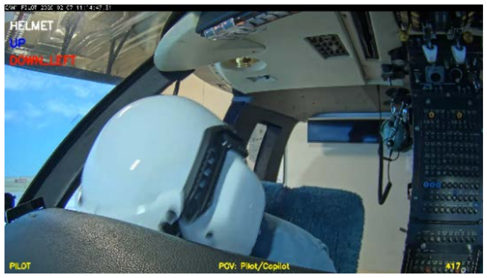

Figure 31c. Pilot frame incorrectly classified by simulator models due to poor representation in the training set.

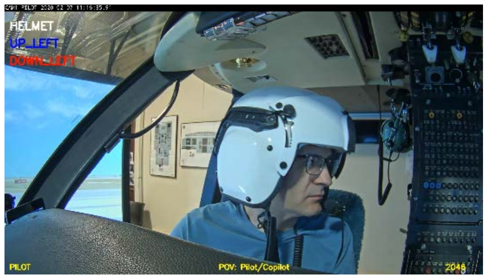

Figure 31d. Pilot frames incorrectly classified by simulator models that are technically incorrect but still present information about the head position.

Figure 31a demonstrates the most common reason for any image-based deep learning algorithm to fail: occlusion. The pilot's hand is adjusting the camera and therefore slightly obscuring the view that the camera has of his head. An occlusion that causes a portion of the image to be blocked will almost always cause issues with a deep learning solution that consists of image data, regardless of the specific application. While the Figure 31a shows an error due to occlusion, Figure 31b shows a misprediction due to image quality. The pilot is moving their head very quickly from left to right and this blurred image causes the algorithm to have a poor headset prediction.

Figure 31c shows the helmet prediction as "Up" when the subject is looking down. This misclassification is due to the limited amount of data available for training. There were most likely not very many images in the training data that resembled this one, so the network will have a difficult time classifying these images in the test set. Figure 31d shows the helmet prediction and combined prediction as "Up_Left" and "Down_Left" when the pilot is just looking left. While these predictions are technically incorrect, it is important to remember that the algorithm was created for estimating head positions so that analysts and accident investigators can interpret the data. That being said, there is still information available in this last image that will describe the general position of the head even if the predictions are not exactly correct.

The issue of occlusion will always be an issue in an image-based deep learning solution. However, the majority of the remaining issues discussed can easily be resolved once more data becomes available. As stated previously, the current datasets primarily consist of simulator data that was created in a controlled environment. When more labeled examples with more variations are included during training, the algorithm should begin to learn more features of the input data and become more generalizable to new data and to specific outlier scenarios.

As an additional experiment, the deep learning models that were trained on simulator data only were used to evaluate the first 10,000 frames of the FAA Flight Dataset. The total number of frames belonging to each of the nine classes is shown in Table 17.

Table 17. Total number of frames belonging to each class in the FAA Flight Dataset.

Class	Total Frames
(0) Down	440
(1) Down_Left	187
(2) Down_Right	402
(3) Left	1348
(4) Right	1155
(5) Straight	4280
(6) Up	1552
(7) Up_Left	333
(8) Up_Right	302

This experiment was conducted to observe the generalizability of the simulator models to a different set of data and observe whether or not the high accuracies from the simulator carry over to the real world. The algorithm performed well on this new data in some cases but the overall performance was much worse than on the simulator test data. This is somewhat of an expected result because the simulator images are different than the real flight video (Figure 19). The confusion matrices (Figure 32a and 32b) for the two nine class networks show these results.

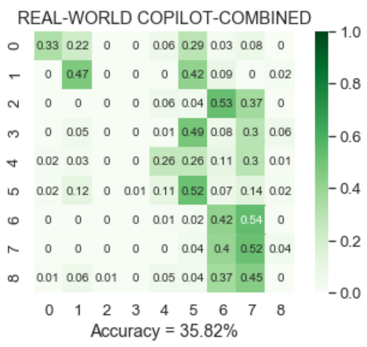

Figure 32a. Combined model confusion matrix for the FAA Flight Dataset evaluated by the simulator models.

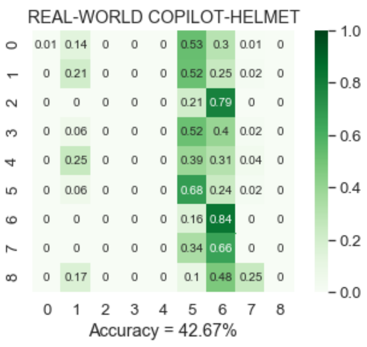

Figure 32b. Helmet model confusion matrix for the FAA Flight Dataset evaluated by the simulator models.

Both models seem to be favoring classes 5, 6, and 7 with almost no predictions in classes 2, 3, 4, and 8. Two examples of incorrectly classified images are shown in Figure 33.

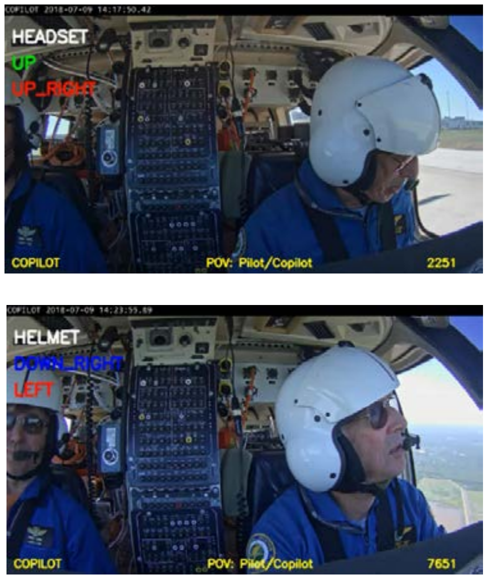

Figure 33. Incorrectly classified images from the FAA Flight Dataset evaluated by the simulator models.

The side classifier was the only model that generalized well to this new data while the head gear classifier and both the combined and headset/helmet models struggled to provide accurate predictions. The change of camera angle and background were most likely the main causes for these decreases in overall accuracy.

Deep Learning Generalized Model Results

To solve the issue of the simulator models poor performance on the real world data, labeled images from the last twenty minutes of the real world flight video were added to the simulator training images using the method described in the, Generalizing to a Real World Dataset section. With this new data available, the four models outlined in red in Figure 34 were retrained with these new images included during training. The CPHeadset_9Class model was not retrained because there was no real world headset image data available.

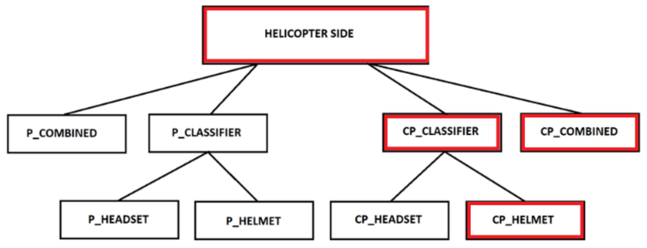

Figure 34. Models that were retrained with real world images included.

The training, validation, and test accuracies of these new models are shown in Table 18.

Table 18. Summary of copilot model accuracies with real world images included during training.

Network (Classes)	Training Accuracy	Validation Accuracy	Test Accuracy
CPHelmet (9)	99.52%	99.65%	99.65%
CPCombined (9)	99.81%	99.81%	99.69%
CPClassifier (2)	100%	100%	100%
HelicopterSide (2)	100%	100%	100%

These results show that both nine class networks performed with about the same accuracy compared to when they were trained and tested on the simulator data only. The helicopter side model and head gear classifier model were both 100% accurate in all cases and this can be attributed to the fact that the features of the image that depict the correct helicopter side and depict the difference between a helmet and a headset are easy to learn.

With the addition of these real world copilot images to the training data, the models were able to generalize much better to the first 10,000 frames of the FAA Flight Dataset. The combined model had an overall accuracy of 91.49% and the helmet model had an overall accuracy of 84.18%. The confusion matrices for these two models are shown in Figure 35a and 35b.

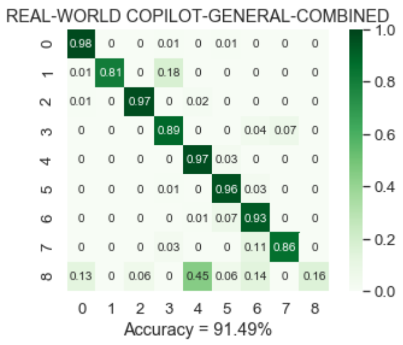

Figure 35a. Combined confusion matrix for the FAA Flight Dataset evaluated by the generalized models.

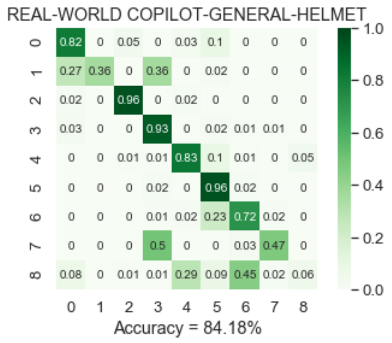

Figure 35b. Helmet confusion matrix for the FAA Flight Dataset evaluated by the generalized models.

The combined model with real world data outperformed the simulator-only model by about 55% and the new helmet model saw an increase in accuracy of about 40%.

While both models do well at classifying images from most classes, they also both struggle to classify images belonging to class 8. Similar to before, this was caused by lack of data in this class. From the entire 30976 images from the real world copilot video, only about 600 fell into class 8. For that reason, the network has a hard time classifying these images simply because it didn't see many images from that class during training. A few examples of correctly classified images are displayed in Figure 36.

Alongside more accurate head pose predictions, the new head gear classifier had no trouble correctly detecting a helmet in all 10,000 test images. By adding just a few labeled images from the real world dataset to the training data, the models were able to generalize well even though the majority of the training images were created in the simulator.

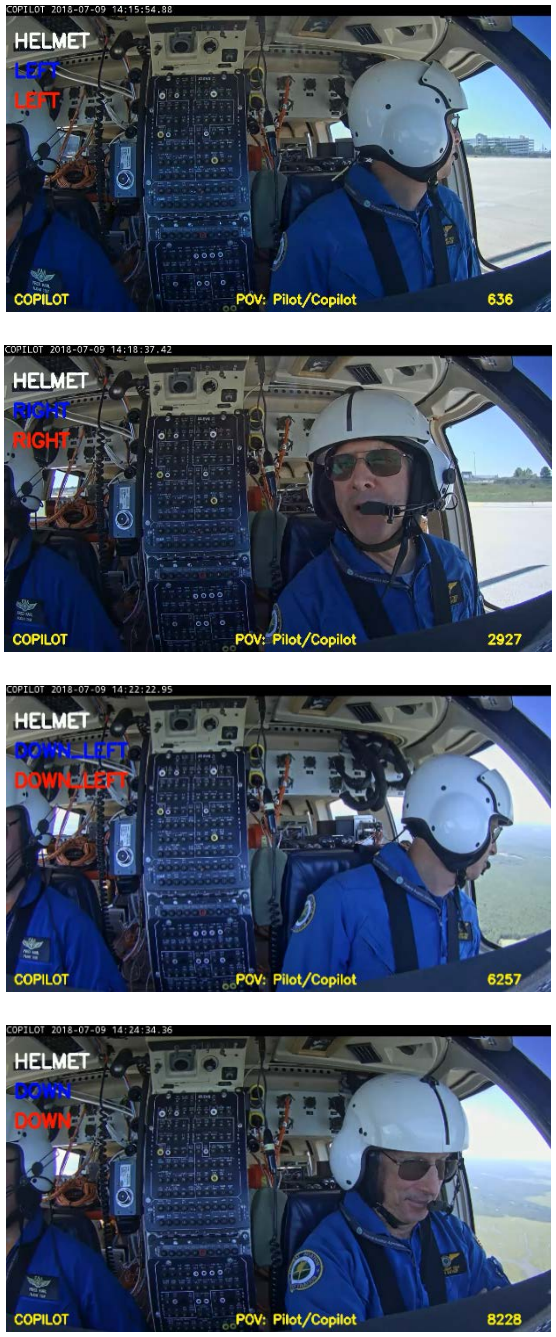

Figure 36. Correctly classified images from the FAA Flight Dataset evaluated by the generalized models.

Comparison of Hybrid Algorithm and Deep Learning Algorithm

In order to compare both algorithms, the quantifiable accuracies are summarized in Table 19.

Table 19. Final accuracies of both algorithms on the FAA Flight Dataset.

Algorithm Name	Accuracy
Hybrid Algorithm	46.01%
Hybrid Algorithm with Compensator	55.26%
Simulator-Only Deep Learning Algorithm	35.82%
Generalized Deep Learning Algorithm	91.49%

The algorithm that performs the best is clearly the generalized deep learning algorithm with an accuracy of 91.49%. However, it is interesting to point out that the deep learning algorithm without the real world data included during training actually performs worse than both versions of the hybrid algorithm. This continues to emphasize the point that a large amount of labeled data that adequately represents the test data is required for the deep learning algorithm to perform well on real flight videos.

If a sufficient amount of labeled data is available, there is no doubt that the deep learning solution will provide the most accurate head pose predictions when compared to the hybrid computer vision algorithm. However, creating a ground truth dataset or labeling any real world flight data can be challenging in this specific application because it is a process that is difficult to automate and will mostly need to be done by hand. This process can be very time consuming and also introduces the potential for human error. The added time needed to create labeled data is well worth the overall accuracy provided by the deep learning algorithm.

On the other hand, the main benefit of the hybrid algorithm is that it does not require ground truth data in order to output head pose classifications. This makes the hybrid algorithm a possible solution if time is more important than accuracy. The hybrid algorithm does take some time to calibrate to each video but this time is negligible when compared to the amount of time it takes to manually label thousands of images and retrain a deep learning model. Since this head pose information will be used for incident/crash analysis however, it is very important that the algorithm provide accurate predictions in all situations, especially in the extreme angle case. For that reason, the generalized deep learning algorithm is the best choice when looking for accurate head pose predictions in the presence of excessive cockpit background, extreme head positions, and added noise from the pilot's operational equipment.

CONCLUSIONS

The goal of this research was to automate post flight video processing and provide safety analysts or accident investigators with data on where a pilot was focused during any particular moment of any given flight. In order to get more helicopter operators to participate in FDM programs, and therefore provide more information for analysts, the video processing needed to be kept simple while being able to have a low cost method of implementation. These issues were both solved using a deep learning algorithm whose only input was video data. The results show that a combination of deep learning models were trained to identify not only pilot/copilot head positions, but also information about the pilot/copilot's head gear and which side of the cockpit the video took place. The overall accomplishments of this research are listed below:

1. Multiple methods for estimating head pose were explored including a hybrid algorithm that utilizes both computer vision and deep learning techniques, and a purely deep learning algorithm that uses a total of nine deep learning models to gain information about head positions, pilot/copilot equipment, and cockpit side.

2. A hybrid head pose estimation algorithm was created that uses classical computer vision techniques of face detection, facial landmark annotation, and the pinhole camera model to calculate angles for classification. This algorithm performed well for frontal facing poses but struggled to classify head positions at extreme angles. However, a compensation method was introduced that aided to increase the number of correct classifications at extreme angles. This algorithm had its limitations but was the main driving force for creating a labeled ground truth dataset for a purely deep learning approach.

3. Working closely with the FAA, a dataset consisting of just under 200,000 labeled images was created in a Sikorsky S76D simulator for helicopter pilot and copilot head positions. The dataset consisted of nine total classes covering the full range of head poses and was used to train the deep learning models for the purely deep learning algorithm. Up to this point, there was no labeled head pose data for helicopter pilots available, and this FAA Simulator Dataset is one of the major contributions of this thesis.

4. A total of nine models were successfully trained using the FAA Simulator Dataset. A two class model was trained to learn what side of the cockpit the video took place. Four networks were then trained for both the copilot and the pilot. A single two class network was trained to learn whether the pilot/copilot was wearing a helmet or headset. On top of that, three nine class networks were trained for predicting head pose. The first model was trained on helmet and headset images combined, the second model was trained on headset images only, and the final model was trained on helmet images only. Each frame of the test video has four accompanying predictions: helicopter side, headgear classifier, combined head pose prediction, and headset/helmet prediction. The separate helmet and headset models are used as added robustness to the combined model since data is limited at this time. This algorithm structure was able to correctly classify head positions for 91.49% of images from the first 10,000 frames of a real world flight video.

5. The main disadvantages of the purely deep learning algorithm are the amount of data required to train a working model, and the time required to collect the labeled data. It can be very difficult to collect labelled data automatically which

results in the majority of data being collected by hand. Although time can be a factor, the main advantage of the deep learning solution is that it can easily be improved as more data becomes available. Once enough data is collected with enough variations, the deep learning models will eventually be able to generalize to any real world flight video.

Future Work and Research Recommendations

As with any deep learning model, improvements can be made as more data becomes available. That being said, the current state of the models can be used to semi-automatically label new test videos to gather more training data. Once more data is collected, the new images can be added into the existing dataset and a new model can be trained.

Using the process of transfer learning would allow for new models to be initialized with the same weights of the current version of the model. This will drastically reduce the amount of time it will take for the network to learn the new features from the newly added images. Rather than relearning the early level features of all the images, transfer learning would allow the network to begin learning the more complex features of the new images right away.

In addition, as more data becomes available, the less necessary the separate helmet/headset models become. At the time of this research, the amount of available data was limited so the added headset/helmet models provide some robustness and confidence to the combined head pose model. However, once the amount of data becomes sufficiently large and the network is able to generalize well to all types of helicopter images, these additional headset/helmet models should be removed to cut down on processing time and to simplify the output of the algorithm. Rather than having four predictions per video frame, the network eventually will only need to output which side of the cockpit the video is on, and a single nine class head pose prediction.

Author Contact:

Eric Feuerstein feuerstee0@students.rowan.edu
Dr. Ghulam Rasool rasool@rowan.edu
Dr. Nidhal Bouaynaya bouaynaya@rowan.edu
Dr. Ravi Ramachandran ravi@rowan.edu
Charles Johnson Charles.C.Johnson@faa.gov

ACKNOWLEDGMENTS

This work was supported by the Federal Aviation Administration (FAA) Cooperative Agreement Number 16-G-015 and NSF Awards OAC-2008690 and DUE-1610911. The authors would also like to acknowledge the support from NVIDIA Corporation in the form of a graphical processing unit (GPU).

REFERENCES

1. National Transportation Safety Board 2017-2018 Most Wanted List of Transportation Safety Improvement, 2017-2018.
2. Verna, B. "Flight Data Monitoring Systems and Non-Required Safety Enhancing Equipment," HEMS Conference, Federal Aviation Administration, 2009.
3. Payan, A., Gavrilovski, A., Jimenez, H., and Mavris, D., "Improvement of Rotorcraft Safety Metrics Using Performance Models and Data Integration," Journal of Aerospace information Systems, Vol.14, No. 1, pp. 26-39, DOI: 10.2514/1.I010467.
4. Murphy-Chutorian, E., and Trivedi, M. M., "Head Pose Estimation in Computer Vision: A Survey," PAMI, 31(4):607-626, 2009.
5. Teulyakov, S., Vieriu, R., Semeniuta, S., and Sebe, N., "Robust Real-Time Extreme Head Pose Estimation," University of Trento Italy, 2014.
6. Sagonas, C., Tzimiropoulos, G., Zafeiriou, S., and Pantic, M., "A Semi-Automatic Methodology for Facial Landmark Annotation," Imperial College London UK, 2013.
7. Ruiz, N., Chong, E., and Rehg, J., "Fine-Grained Head Pose Estimation Without Keypoints," Georgia Institute of Technology, 2018.
8. Gourier, N., Hall, D., Crowley, J. L., "Estimating Face Orientation from Robust Detection of Salient Facial Features," Proceedings of Pointing, ICPR, International Workshop on Visual Observations of Deictic Gestures, Cambridge, UK, 2004.
9. Mallick, S., " Head Pose Estimation Using OpenCv and Dlib," Availabe: https://www.learnopencv.com/head-pose-estimation-using-opencv-and-dlib/
10. Hata, K., Savarese, S., "Course Notes 1: Camera Models," CS231A: Computer Vision, From 3D Reconstruction to Recognition, Stanford University, Stanford, California., Mar., 2018.
11. Slabaugh, G., "Computing Euler Angles from a Rotation Matrix", unpublished.
12. Ng, A., Class Lecture, Topic: "Neural Networks and Deep Learning," Deep Learning Specialization, Coursera, Online. Accessed, Aug., 2018.
13. He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.
14. Szegedy, C., *et al*., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Patter Recognition (CVPR), Boston, MA, 2015, pp. 1-9.
15. Chollet, F., "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Patter Recognition (CVPR), Honolulu, HI, 2017, pp. 1800-1807.