Max–Planck–Institut für biologische Kybernetik
Max Planck Institute for Biological Cybernetics

Technical Report No. 196

Helilab User Manual: Human Behavior and Flight Data Acquisition and Analysis

Jung-Jin Lee[1] and Heinrich H. Bülthoff[1]

May 2012

This Technical Report has been approved by:

Director at MPIK	Postdoc at MPIK

Max–Planck–Institut für biologische Kybernetik
Max Planck Institute for Biological Cybernetics

Technical Report No. 196

Helilab User Manual: Human Behavior and Flight Data Acquisition and Analysis

Jung-Jin Lee[1] and Heinrich H. Bülthoff[1]

May 2012

[1] Department Bülthoff, email: jung-jin.lee;heinrich.buelthoff@tuebingen.mpg.de

This report is available in PDF–format at http://www.kyb.tuebingen.mpg.de/fileadmin/user_upload/files/publications/2012/tr-196.pdf. The complete series of Technical Reports is documented at: http://www.kyb.tuebingen.mpg.de/techreports.html

Helilab User Manual: Human Behavior and Flight Data Acquisition and Analysis

Jung-Jin Lee and Heinrich H. Bülthoff

Abstract. We, here, introduce a flight simulator dedicated to collect both human behavior and flight data with high fidelity. This paper illustrates the current settings of the flight simulator, key issues in using the simulator, and the standard procedure of data acquisition and analysis.

---

1 Introduction

This paper describes the settings of a flight simulator established in Helilab (located in Spemannstraße 38[1]) designed to investigate human behavior in cockpit scenarios[2]. The flight simulator is designed and deployed to allow participants the freedom to perform actions similar to the real-world environment with high visual fidelity during recordings[3]. The current setting of a flight simulator in our laboratory allows us to measure both eye movement behavior and bio-signal of participants, but this paper focus on human eye movement behavior and flight data collection and analysis without any information of bio-signal acquisition and processing. This paper illustrates the layout of equipments and its details in section 2 and 3. Issues considered in data acquisition and analysis are described in section 4. The procedure of data acquisition and analysis are illustrated in section 5 and 6, and a simple demo is shown in section 7. The source codes for data analysis are attached in appendix A, and supporting materials are listed in appendix B.

2 Requirements

First, the experimental set-up should allow the participant the freedom to perform actions similar to the real-world environment. For example, his movements should be unrestrained. This allows findings in the laboratory to be generalizable to the real-world, as well as to minimize irrelevant movement artifacts in the data. Second, it is important to provide high visual fidelity during recordings. This would ensure that the user is referencing the appropriate information with his eye-movements. Finally, it is crucial to formally define the task-routines under investigation, in terms of the specific actions and visual information that are expected to be necessary for successful task completion. Failing to do so could yield eye-movement data that is difficult to interpret meaningfully and in a principled fashion.

Thus, a flight simulator was established in our laboratory with the following attributes;

- realistic real-time simulation of a world environment
- large field-of-view
- unrestrained gaze tracking
- configurable control interface

In section 3, we illustrated the specification of current settings in Helilab considering the requirements above.

[1] Helilab (room no. 026) is locate at the inside of Panolab (room no. 025) on the ground floor.

[2] The current work is performed within the context of myCopter, a project funded by the European Union under the 7th Framework Programme to research technologies that will enable the use of personal aerial vehicles for daily commuting [1]. One part of this project focuses on measuring and modeling behavior of a *consumer pilot* in active flight control tasks.

[3] Details in the requirements and specification of a flight simulator in Helilab are illustrated in section 2 and 3.

2.1 Additional requirements for data processing

All source codes for data analysis are written in MATLAB®. The followings are the requirements in using MATLAB® source code provided with this paper:

- MATLAB® R2010b or any release above R2010b
- Database toolbox
- MySQL™

(a) A flight simulator with a large field-of-view

(b) Cockpit side view

(c) Cockpit front view

(d) Two eye-tracking devices

Figure 1: A flight simulator with eye-tracking devices

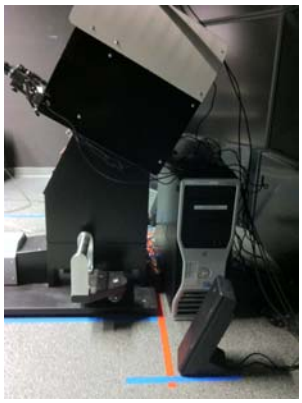
3 Specification

3.1 Realistic real-time simulation of a world environment with a large field-of-view

Eye-movement research is ideally conducted in an environment that affords high visual fidelity. In our set-up, nine screens (each 102 x 57 cm; 1366 x 768 pixels) comprised a multi-panelled display that afforded a field-of-view of up to 105° x 100° visual angle (see Fig. 1(a)). In addition, a dedicated screen provided the user with a heads-down flight instrument panel. Graphical rendering[4] on all displays were performed simultaneously with clusters of 10 graphics workstations (see Fig. 2). We employed a flight simulation software (Flightgear) [2] to simulate the flight dynamics of a Bo105 helicopter[5] to compute the appropriate graphics for the external world environment and the flight instruments.

3.2 Unrestrained gaze tracking

Two remote eyetrackers[6] (faceLAB^{TM}5; SeeingMachines) are deployed to record eye-movements on the flight instrument and front displays (see Fig. 1(d)). This system computed the gaze vector of the user in the world-environment and, from this, recorded the intersection of the user's gaze with the flight instrument panel at 60 Hz. Eyetracking could be performed to the accuracy of 1.5°, which was sufficient for determining the specific flight instrument that was referred to — that is, fixated by the user's gaze — at any given time. In addition, a dispersion-threshold identification algorithm was applied to the raw eye-movement data to specify the duration and location of meaningful fixations and to remove artifacts [3]. Namely, we removed eye-movements[7] that were faster than 60 $^\circ$/sec and short fixations with a duration of less than 150 msecs.



(a) A graphic workstation deployed behind a cockpit

(b) Nine graphic workstations (vertical stack) and two additional workstations for eye-tracking devices

(c) Time synchronization server

Figure 2: Clusters of workstations for visualization, eye-tracking, and time synchronization

4 Issues in Data Acquisition and Analysis

In the current settings of our flight simulator in Helilab, we can collect three data simultaneously; eye movement behavioral data (from two set of eye-tracking devices), control input data (from two joysticks for position and throttle control and a set of rudder paddle sending input signals to Flightgear), flight data (from Flightgear). Eye movement data is collected from two workstations linked to each eye-tracking device, and control input and flight data are collected and can be retrieved from a workstation with Flightgear in it. As several independent systems collect different types of data, time synchronization is crucial prior to data acquisition. The method for time synchronization is illustrated in section 5.1, and the procedural steps of each data acquisition for human behavior

[4]Details are illustrated in appendix B-3. Multiple Monitors in Flightgear.

[5]The flight model can be easily changed to the other model based on the purpose of an experiment.

[6]For more information, please visit the website listed in appendix B.

[7]The criteria to identify fixations can be adjusted by changing parameters in MATLAB® source code given in appendix A.

data (i.e., eye movement data) and flight data are duly described in section 5.2 and 5.3.
Remaining issues are about data analysis. Time series data of eye movement and flight simulation are restored in a special format as given from the vendor of commercial products. The vendor-dependent format can be converted to .txt file for convenience in raw data analysis, but it is not an efficient way of observing and analyzing data collected. Therefore, we developed a data management program with a standard procedure of data collection using MATLAB® and MySQL™ which is a well-known and widely-used open-source database management system. Details are illustrated in section 6.

5 Data Acquisition

Knowing the system architecture of a flight simulator deployed in Helilab will help you to understand contents illustrated in this section. As shown in Fig. 3, multiple individual systems are set for different purpose. Whole system architecture can be divided into two categories; visualization and data collection. Ten graphics workstations are dedicated to visualize scenery during flight simulation, and two workstations are dedicated to collect eye movement data. One of the graphics workstations both work in visualizing a control panel and collecting flight data. Human eye movement data and flight data are collected in different independent systems, so the time stamps of each data have to be synchronized. Time synchronization server is dedicated in time synchronization using NTP FastTrack[8] and the detail information on how to synchronize clocks of each workstation is described in the following subsection below. The standard procedures of data acquisition for human behavior data and flight data are also illustrated in this section.

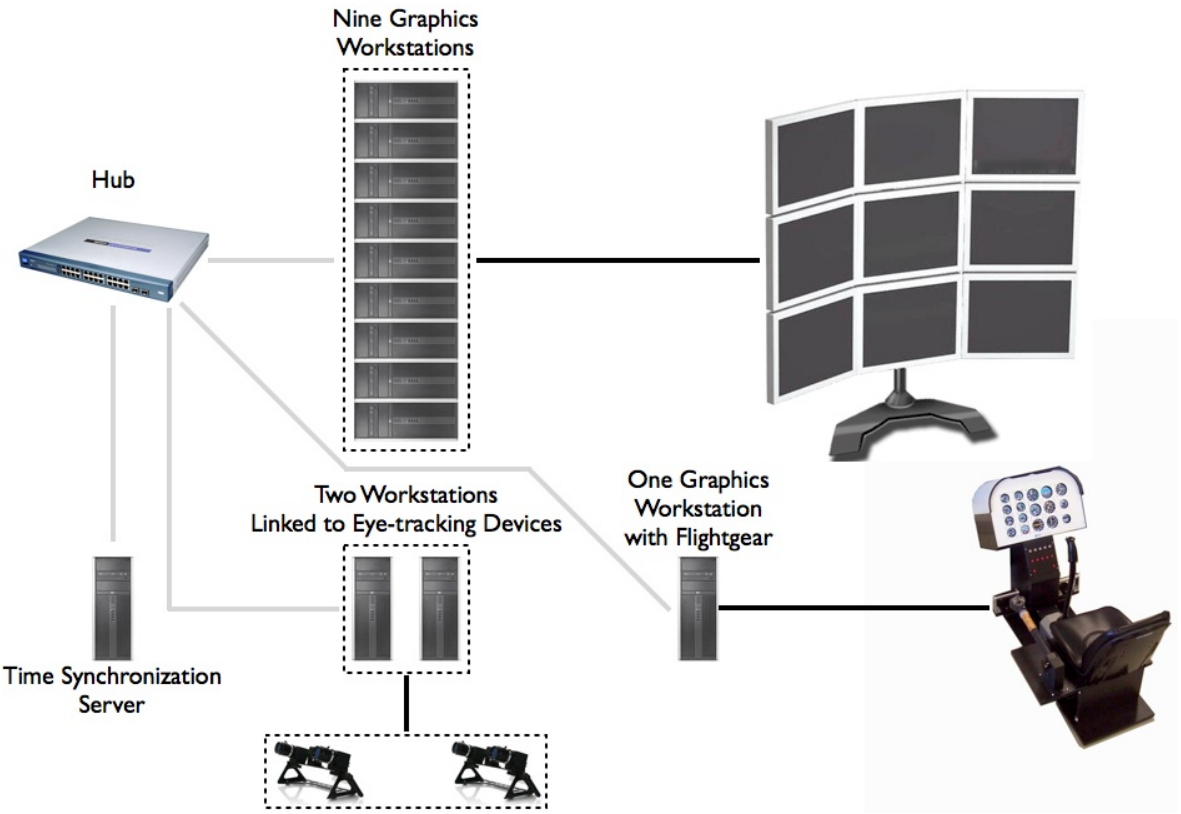
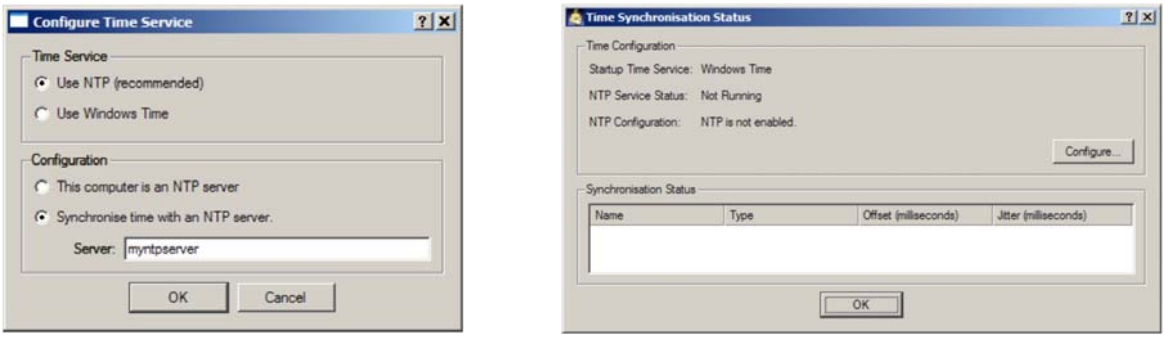


Figure 3: A conceptual system architecture of a flight simulator in Helilab; Black lines represent I/O connection between computer and displays and/or input devices. Gray lines show network connectivity among server and workstations. (Note: this is a conceptual system architecture, so the physical organization of devices in a cockpit simulator can be different from the figure given.)

[8] See, appendix B. You can also download official manual of NTP FastTrack via URL given.

5.1 Time Synchronization

Time synchronization can be easily achieved by using NTP FastTrack. You can download the software via URL given in appendix B.2, and you also can download the manual of NTP FastTrack (see appendix B.3). We briefly illustrate issues need to be considered in using NTP FastTrack to synchronize clocks of each computers.
The first thing you have to do is to download NTP FastTrack program and install the program on each computer you want to synchronize. In our settings, we basically need to install the program in three workstations; two workstations linked to each eye-tracking device, and one graphics workstation has Flightgear in it. Then you need one additional computer as a NTP server. You might think that it would be possible to use one of three computer as a NTP server, but it is not a right way to set-up NTP synchronization program. During an experiment, three workstations constantly collect human behavior and flight data which definitely require a lot of computing resource. If you set one of the computer as a NTP server, the lack of computing power will highly affect to both synchronization duration and the accuracy of time synchronization, and you will never get synchronized time stamp with minimum jitters which is necessary in our experimental purpose. It would be a wise decision to make an additional NTP server only dedicated in time synchronization.



(a) Configuration window

(b) Status window

Figure 4: NTP FastTrack configuration and status monitor (adapted from "Guide to Use NTP FastTrack" by SeeingMachines)

Since you have a server and workstations to sync, then you need go to configuration menu of NTP FastTrack and specify whether a machine is an NTP server or not (see Fig. 4(a)). You also need to set NTP FastTrack to use Windows time not NTP since the flight simulator is set as a close-network only accessible via a network hub device deployed in Helilab. Since you set whole NTP FastTrack programs in each machine, then you need to wait until jitter is minimized (see Fig. 4(b)). According to the manual given by SeeingMachines (see, appendix B.3), it takes around 24 hours for initial time synchronization, but it took only several hours to be stabilized in our experience. During recording session, time clocks on each computer have to be synchronized, so you should have to check the status monitor shown in Fig. 4(b) on each computer prior to your experiment.

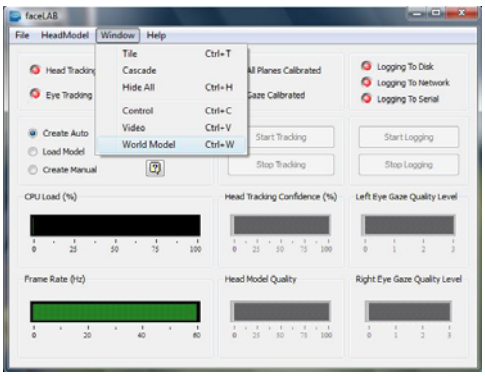
5.2 Human Behavior Data Acquisition

In this section, we describe about how we setup devices to observe eye movement of participants and what is the standard procedure to collect human behavior data.

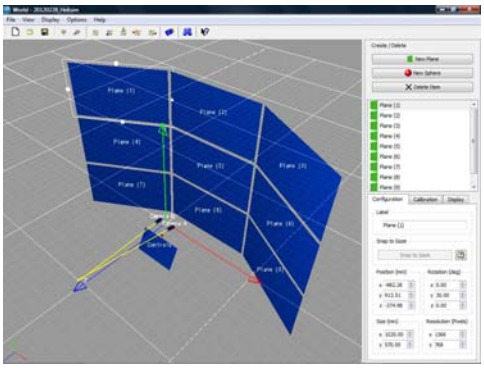
5.2.1 Settings for Human Behavior Data Acquisition

Following instructions illustrated in a manual offered by SeeingMachines (see appendix. B.3), you setup your eye-tracking system. The first step is to input coordinate information of two eye-tracking devices in faceLAB software (see appendix. B.1). As shown in Fig. 5, you can set relative coordinates of 10 displays from each eye-tracking device. The coordinate information is the physical basis of processing eye movement, so the information has to be accurate as possible. In our flight simulator, we physically measured the location of each device using a laser distance meter (see Fig. 6).
Remaining issue is that the coordinate information is about relative distance between each eye-tracking device and displays, so the information have to be updated whenever the location or position of eye-tracking devices are changed. Although, you did not move eye-tracking devices, the coordinate information of each eye-tracking device can be changed by the body movement of participants or incidents. Therefore, we need more reliable solution to

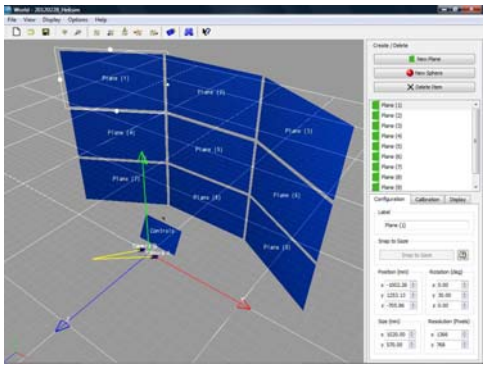
update coordinate information; updating information by physically measured relative distance among devices prior to each recording session would be exhaustive.



(a) World Model menu enables to open built-in Worldview program in faceLAB



(b) World Model settings of an eye-tracking device deployed on the top of an instrument panel



(c) World Model settings of an eye-tracking device deployed on the bottom of an instrument panel

Figure 5: World Model settings with built-in Worldview software

(a) A flight simulator (color lines on the floor are used to calculate coordinate information of each device)

(b) A color line assists to position a monitor deck on the left side (pointed with a yellow arrow)

(c) A color line assists to position a monitor deck on the right side (pointed with a yellow arrow)

Figure 6: The organization of devices in a flight simulator

The solution we propose is using a fixed reference point based distance measure. We set a reference point on the top of an instrument panel display, and it is used as the pivot point of physical distance measure (see Fig. 7 and 8). There are four important coordinate information used to measure distance from the reference point; the center point of each eye-tracking device, the center point of a top-center display, and the center point of an instrument panel display. Using fixed relative distance from the top-center display, you can calculate relative distance from the reference point to 9 displays. In this settings, the relative distance from the reference point to 10 displays are

fixed since a cockpit module is not moved. Since you match the position of the cockpit module to lines drawn on a floor, you can get an accurate physical measure. However, you still need to calculate the relative physical location of the 10 displays w.r.t. the position of eye-tracking devices. For your convenience, we made a simple spreadsheet using Microsoft® Excel (see appendix. B.2 WorldModelMaker). You only need to measure an angular distance information from two eye-tracking devices from the reference point (current World Model for each eye-tracking devices are given in appendix. B).

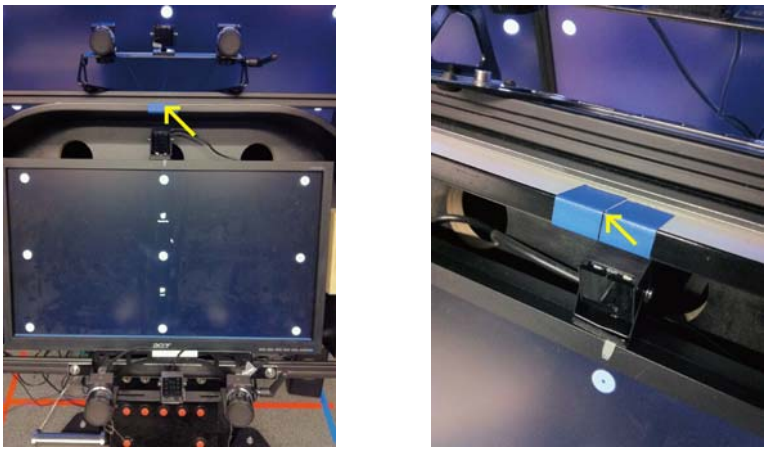
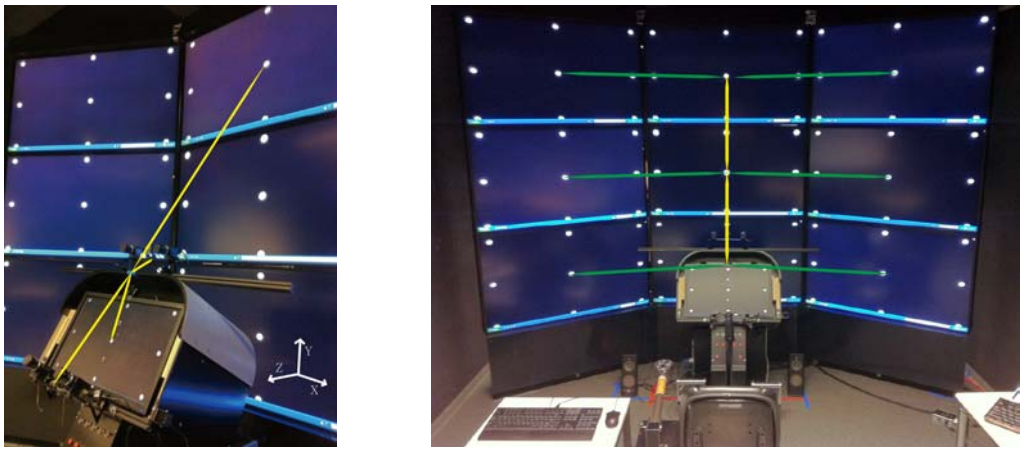


Figure 7: A reference point for physical space measure for devices (pointed with a yellow arrow)



(a) Relationship between a reference point and key devices; the length of yellow lines represent the distance from major four devices to the reference point.

(b) Relationship among displays; the length of yellow lines represents the distance from top-center display to center-center and bottom-center display, and the length of green lines represents the relative distance among displays on a same horizon.

Figure 8: Relationship among devices in three-dimensional space

5.2.2 Recording Procedure

The procedure of eye movement data acquisition is duly described in faceLab user manual (see appendix. B.3). There are two addition issues we need to address in this section; SID calibration[9] and faceLAB LINK. Without SID calibration, eye movement data collected by using faceLAB will be fully relied on physical measurement which

[9]Background images for SID calibration are provided in appendix. B.4

Figure 9: Moving direction of each eye-tracking device

would not be sufficient enough to get accurate data. Therefore, you should conduct SID calibration for displays[10] (see Fig. 10). SID calibration has to be done prior to each session of recording to get maximum accuracy.
Another issue is about the usage of faceLAB LINK. faceLAB LINK is a software to collect more accurate data in large visual field covered by multiple eye-tracking devices (see appendix. B.3). However, we do not recommend to use faceLAB LINK in your experiment in Helilab. faceLAB LINK is designed to give better precision during eye movement retrieval with multiple eye-tracking devices, but it seems only works with a single large screen or multiple displays consists a single plane. In our experience, faceLAB LINK does not guarantee us to collect eye movement data with sufficient level of accuracy. We recommend you to independently collect human eye movement behavior data using two eye-tracking devices; one (deployed on the bottom of an instrument panel display) for collecting eye movement data on 9 displays in front of a cockpit, and one another device for acquiring data on an instrument panel.

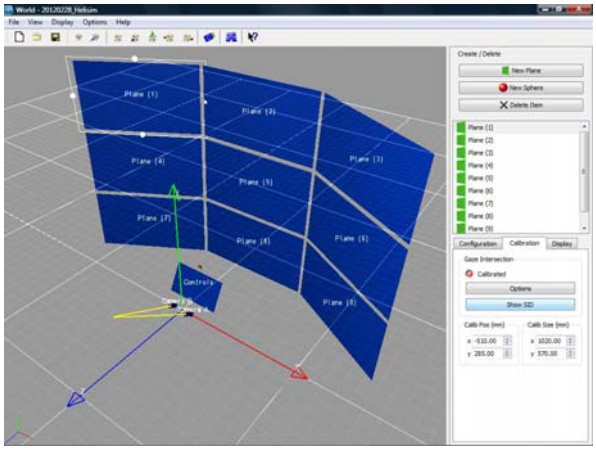


Figure 10: Calibration menu on built-in Worldview software

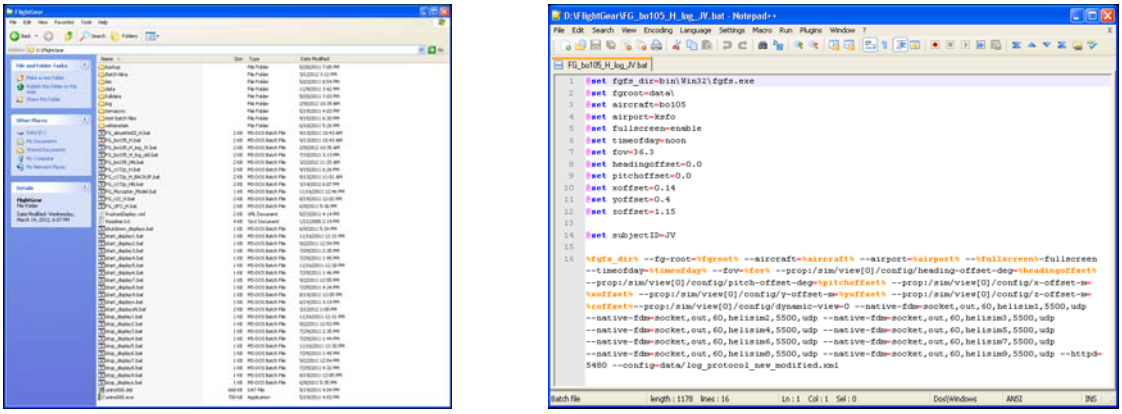
[10]In our experience, it is required to conduct SID calibration on center-center display with an eye-tracking device deployed on the top of an instrument display, and one additional SID calibration on the instrument panel display with an eye-tracking device deployed on the bottom of an instrument display.

5.3 Flight Data Acquisition

In this section, we describe about how we setup a flight simulation program (i.e., Flightgear) to collect flight data and the procedure of recording in a flight simulator.

5.3.1 Settings for Flight Data Acquisition

There exist two ways to collect data from Flightgear simulation. The first method is to record flight data by clicking record menu after starting simulation, and the second way is making a batch file with a XML file enables us to collect data automatically whenever we start Flightgear. In current version of Flightgear, you only can record less than 10 parameters from the simulator since you use the first way. The advantage of the second way is that you can record as many parameters as you want by scripting XML file contains the list of parameters you want to record. We recommend you to use the second way, and you need to make one batch file and one XML file to use the second way of recording flight data.[11]



(a) Flightgear folder (b) A batch file with the location information of a XML file

Figure 11: The location and the script of a batch file

As shown in Fig. 11, you have to make a batch file. In the end of the batch file, you need to specify the name of a XML file which the batch file will be used as shown in Fig. 11(b). The next step is to make a XML file which contains the list of parameters of interest[12]. As shown in Fig. 12, the location of a XML file has to be match to the information written in the batch file illustrated above. The last step is to make folder to restore a log file. The folder name and location are pre-defined in your batch file, and you need to manually make the folder in a right location (see Fig. 13). Flightgear starts to record flight data in the log file since you start recording by clicking the batch file you made, and the log file will be saved inside of the pre-defined folder as shown in Fig. 13(b).

5.3.2 Recording Procedure

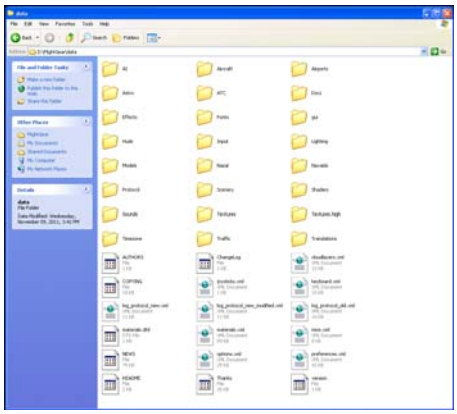
Since you successfully made two files as illustrated above, it is very simple to collect flight data. You only need to click the batch file you made, and Flightgear will start to run flight simulation and record flight data. Recording will be automatically stopped since you switch off Flightgear.

6 Data Analysis

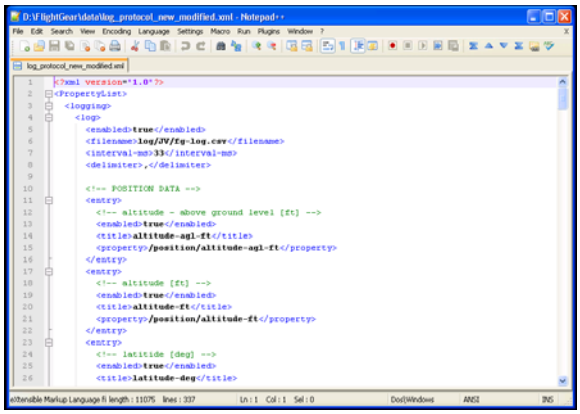
Since you collect data from multiple recording devices, you need to transform data into desirable form enable to effectively manage and analyze data. Before starting analysis on human eye movement and flight data, we first need to install database management system (DBMS) to restore data. Using database toolbox, we will show how to retrieve data for analysis in MATLAB®. Pre-processing is also required to process raw data prior to transfer the data into DBMS. The routine of data processing is illustrated below.

[11] A batch file and an XML file are provided in appendix B.4

[12] The full list of parameters can be found in Flightgear menu on top of the screen since you start Flightgear.

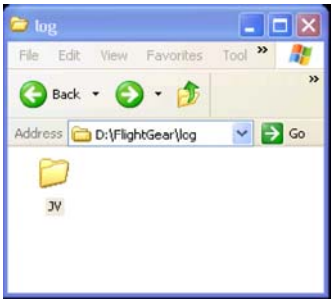


(a) A folder which has a XML file in it

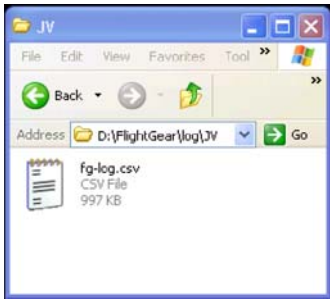


(b) A XML file with the list of parameters of interest

Figure 12: The location and the script of a XML file



(a) A target folder where the log file will be placed in.



(b) A log file which contains flight data

Figure 13: The location of a log file recorded by Flightgear

6.1 Data Process Routine

Fig. 14 shows the routine process of data transform and transfer. Details are illustrated in section 6.3, and DBMS installation required prior to pre-process is described in section 6.2.

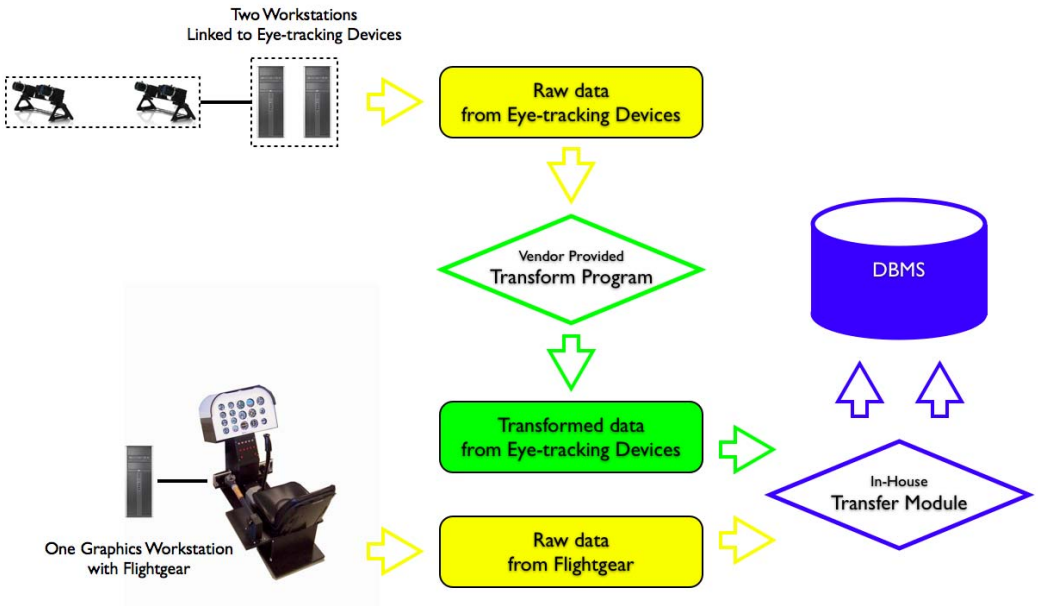


Figure 14: Data Process Routine

6.2 Settings of Database

The section is about installation of MySQL™. More details can be found in the official website of MySQL™, and here we briefly describe about the installation. First, you need to install MySQL™ (see appendix B.2). You do not need to change any settings during installation but follow default setting steps. Make sure to remember user identification (ID) and password for MySQL™. The second step is to create a database. You need to open MySQL™ console and login. Using SQL command 'create database', you can create a new database (see Fig. 15). Remember the name of newly created database (in our settings, we named the database 'mycopter'). This will be used in the next step of ODBC setting.

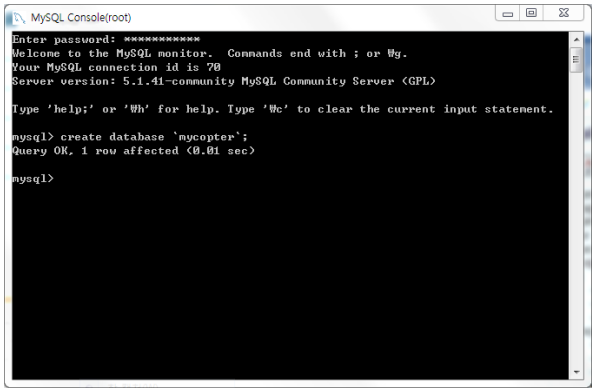
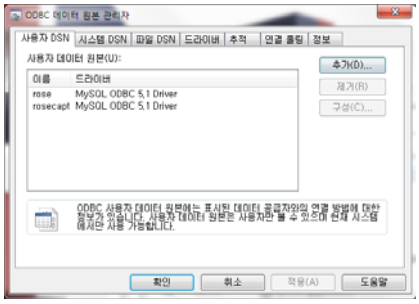
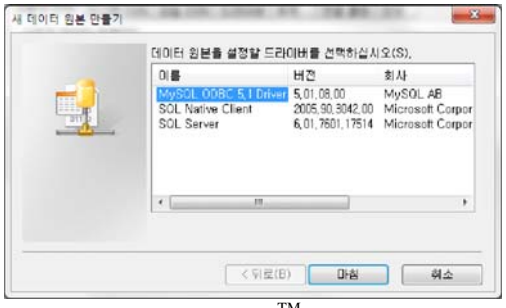


Figure 15: Create a new database using MySQL console

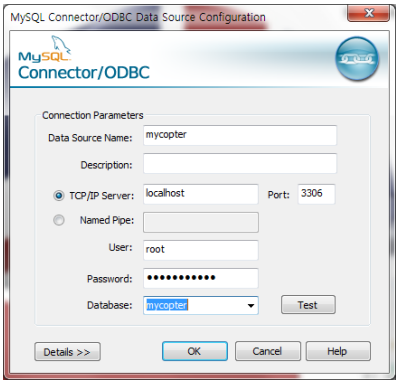
The third step is to install MySQL™ ODBC connector (see appendix B.2). ODBC connector enables us to access DBMS. Download and install MySQL™ ODBC connector. After installation, you need to set ODBC in Windows' control panel. By clicking 'start' button of Windows, you will find a single line command window for search programs and files on top of 'start' button. Since you type 'ODBC', you will find a program located in 'Control Panel' called 'Set up data sources (ODBC)'. Run the program, and you need to setup ODBC (steps for setting a new ODBC connection are illustrated in Fig. 16).



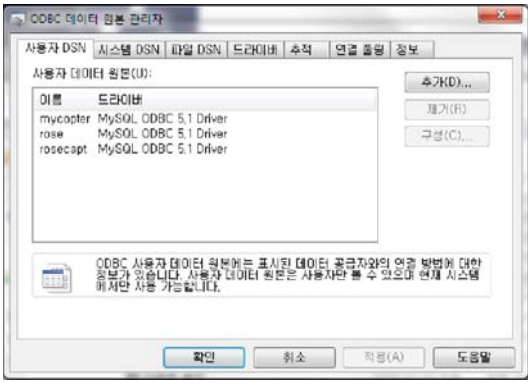
(a) Step 1: start a new ODBC setting by clicking 'Add (D)' button



(b) Step 2: select MySQL™ ODBC connector



(c) Step 3: input relevant information to setup ODBC settings



(d) Step 4: new ODBC setting is created

Figure 16: ODBC Settings

6.3 Pre-processing

Pre-processing consists of two parts (see Fig. 14). The first part is converting raw eye movement data collected by using converting programs provided by SeeingMachines. Since you record participants' eye movement data from individual eye-tracking devices, you will get 7 different files; 'Eye.fll', 'Face.fll', 'Head.fll', 'HeadV2.fll', 'Timing.fll', 'World.fll' and 'WorldV2.fll' (note: file names can be changed based on the settings of faceLAB™). Basically, you can transform raw data into .txt files using Worldview program, except 'WorldV2.fll' (see Fig. 17). Since 'WorldV2.fll' contains participants' eye movement data, we need additional converting program called 'logconverter' (see appendix B.2). Fig. 18 shows about how to convert 'WorldV2.fll' into .txt file.

Since all transformation are done, now is the time to run an in-house built transfer program called 'myCopter_DataTransfer' written in MATLAB® (see appendix A.1). It automatically transfers all human behavior and flight data into DBMS at once, and you only need to specify the name of .txt files in 'myCopter_DataTransfer.m' file. Detail information is written in the source code with comments.

6.4 Human Behavior Data Analysis

Eye movement data collected have to be processed by multiple functions (programs). You only need to run 'analysis_fixations.m' which runs all related programs; annotation validation, time format converting, data reduction, fixation identification, instrument coordinate mapping and fixation plot.

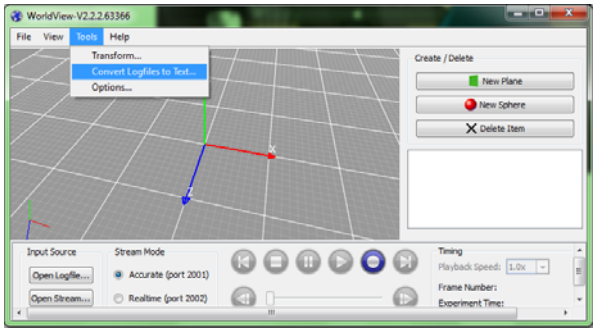
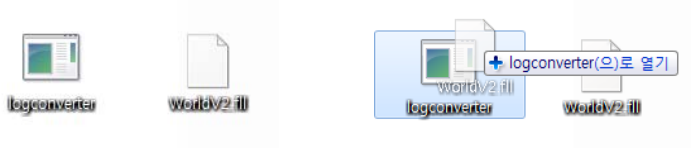


Figure 17: Transform raw data into .txt file



(a) Vendor provided 'logconverter' program with a raw data

(b) Converting raw data to .txt file by dropping raw data on top of 'logconveter' program icon

Figure 18: How to use 'logconveter' program

- 'analysis_fixations'
- 'annotation_validation'
- 'data_reduction'
- 'fixation_identifier'
- 'gmt_serial_converter'
- 'instrument_selection'
- 'plot_fixations'

As the result of processing, you can discriminate fixations from raw data, and you also can use this information to calculate statistics of fixation w.r.t. fixation coordinate on an instrument panel. Detail instructions are illustrated in source codes (see appendix A.2).

6.5 Flight Data Analysis

Flight data collected from Flightgear includes many information including trajectory of an airplane. As illustrated in section 5.3.1, you can record many different parameters from Flightgear as much as you want by modifying a batch file and a XML file. We, here, provide two different simple programs to visualize flight trajectory.

- 'PlotTrajectory_SingleFlightPlan'
- 'PlotTrajectory_MultipleFlightPlan'

These two programs not only show you the trajectory of an airplane with single or multiple flight plans, but also let you know how to use flight data restored in DBMS (see appendix A.3). The figures can be achieved by running two programs will be illustrated in section 7 with designated flight plans as demonstration.

7 Demo: A Flight Plan with Basic Maneuvering Tasks

7.1 Flight Plan Design

We design two sets of flight plan; a simple flight plan with six different tasks and flight plans with three difficulty levels. A simple flight plan is used to demonstrate human eye behavior acquisition and analysis, and multiple flight plans are used to show how to acquire flight data and analyze it.

7.1.1 A Simple Flight Plan with Six Different Tasks

With the assistance of an expert helicopter pilot (ex-Military, 650 flight hours), we developed a flight scenario that consisted of the following basic maneuvering tasks: ascending, descending, acceleration, deceleration and turns. (see Fig. 19)

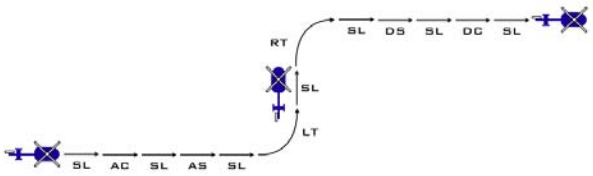


Figure 19: A flight plan; SL: straight and level flight, AC: acceleration, DC: deceleration, AS: ascending, DS: descending, LT: left turn, RT: right turn

Such maneuvers are comparatively easy to execute and are within the capabilities of a non-expert pilot with experience in flight simulators. Our expert pilot recommended the primary flight instrument that he typically relies on for the execution of each basic task (in bold), as well as what he considered to be secondary instruments (see Table 1).

Task	Primary Information	Requirement
Acceleration	Altimeter	800ft
	Speed indicator	70 to 100knots
	Compass	East
Deceleration	Altimeter	800ft
	Speed indicator	100 to 70knots
	Compass	East
Ascending	**Altimeter**	400 to 800ft
	Vertical speed indicator	+500ft/min
	Speed indicator	70knots
	Compass	East
Descending	**Altimeter**	800 to 400ft
	Vertical speed indicator	-500ft/min
	Speed indicator	70knots
	Compass	East
Turn Left	Altimeter	800ft
	Speed indicator	100knots
	Compass	East to North
	Artificial horizon	-15°
Turn Right	Altimeter	800ft
	Speed indicator	100knots
	Compass	North to East
	Artificial horizon	+15°

Table 1: Different Flight Phases; bolded instruments represent the major instrument needed to accomplish each task as defined by an expert pilot.

7.1.2 Flight Plans with Three Difficulty Levels

We design new flight plans with different difficulty levels. Using three designated point as illustrated in Fig. 20, we construct three different flight plans. Detail information of three locations is provided in 'General Information of Three Locations.docx' (see appendix B.4). Tasks consists each flight plan are illustrated in 'Time-Altitude Chart of Flight Plans.pdf' and 'New 3-Column Flight Phases.docx' (see appendix B.4). Annotation is also necessary to separate data based on task, and 'Basic Commands to Record Annotation IDs.docx' and 'Key Combinations Correspond to Task.docx' are ready for the annotation. For your convenience during recording, check list is also given as 'Flight Plan (Experimental Purpose).docx' (see appendix B.4).

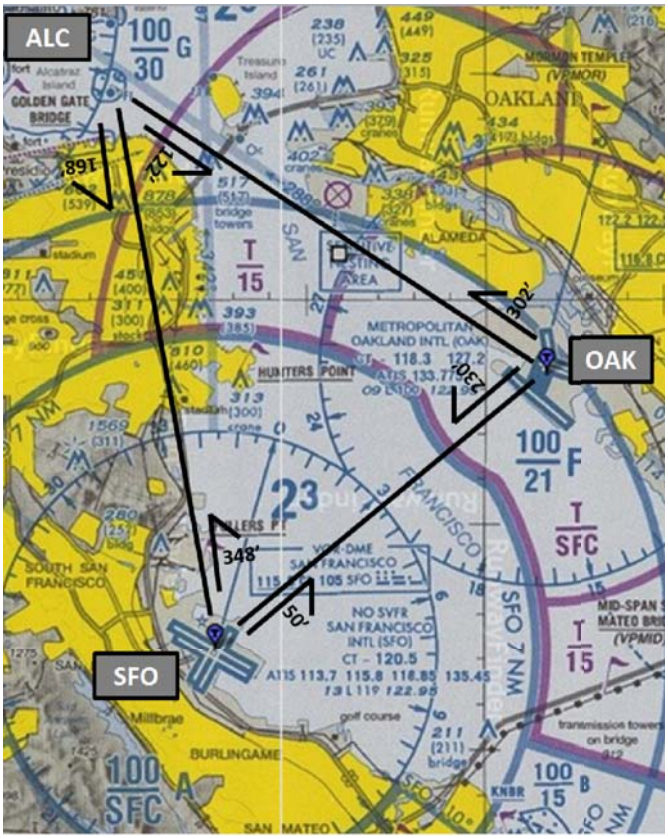


Figure 20: Aeronautical map for flight plans with three difficulty levels

7.2 Data Acquisition

7.2.1 Human Behavior Data Acquisition with a Simple Flight Plan with Six Different Tasks

Human behavior can be divided into two parts in our system; eye movement data and flight control data. Eye movement data can be acquired by using faceLAB™ 5, and flight control data can be collected from Flightgear (see appendix B.3 and section 5.3). The important issue in collecting data from faceLAB™ 5 is annotation. During the recording, you can annotate, and the annotation can be used to separate data based on tasks. You have to make sure how to assign annotation ID for each task (MATLAB® codes provided in appendix A.2 shows how to assign and process annotation ID; each 'straight and level flight' are annotated 1 and each 'designated tasks' are annotated 12). Another issue in annotation is input missing. In our experience, pressing a key or clicking mouse sometimes does not recorded by faceLAB™ 5. By allowing multiple pressing and clicking can help to reduce the probability of losing annotation information. The codes provided in appendix A.2 allows multiple annotation input, so you can click or press a key multiple times for each task.

7.2.2 Flight Data Acquisition with Flight Plans with Three Difficulty Levels

Flight data acquisition is simple. Since you set parameters to log by modifying a XML file, you only need to click a batch file liked to the XML file (see section 5.3.1). When you terminate Flightgear simulation, logging is also stopped automatically.

7.3 Data Analysis

7.3.1 Human Behavior Data Analysis with a Simple Flight Plan with Six Different Tasks

As illustrated in section 6.4, we provided several MATLAB® program to process and visualize fixations during flight maneuvers. Fig. 21 shows the layout of instrument panel, and Fig. 22 shows fixations on the instrument panel by processing raw data using 'analysis_fixations'.

Figure 21: The layout of instrument panel of a Bo105 model from Flightgear; the red circle in the center of the panel represents the 1.5° resolution of the eyetracker (not shown during experiments).

7.3.2 Flight Data Analysis with Flight Plans with Three Difficulty Levels

As described in section 6.5, we provided two MATLAB® program to visualize trajectory of flight. Fig. 23 shows the trajectory of an airplane with each flightplan by executing 'PlotTrajectory_SingleFlightPlan'. If you want to see all trajectories at once, you can use another program 'PlotTrajectory_MultipleFlightPlans' (see Fig. 24).

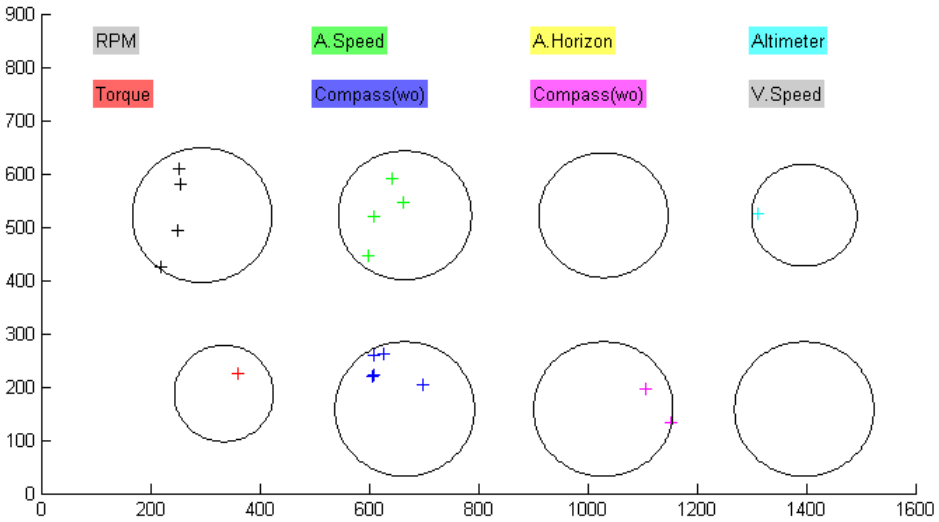
A Appendix: MATLAB® Source Code

A.1 Pre-processing

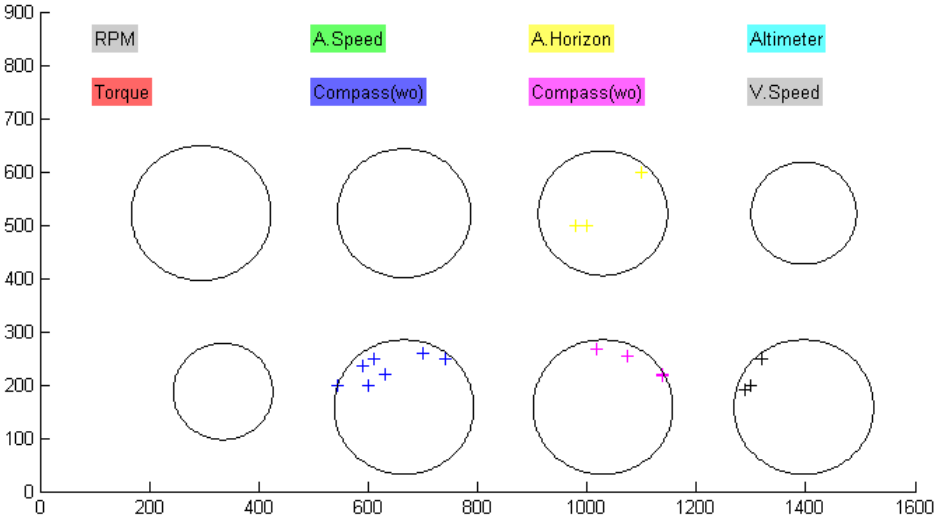
- myCopter_DataTransfer (in-house script)
  `\\uni\home\projects\helilab\myCopter_DataTransfer.m`

A.2 Human Behavior Data Analysis

- analysis_fixations (in-house script)
  `\\uni\home\projects\helilab\annotation_validation.m`
- annotation_validation (in-house script)
  `\\uni\home\projects\helilab\analysis_fixations.m`
- data_reduction (in-house script)
  `\\uni\home\projects\helilab\data_reduction.m`
- fixation_identifier (in-house script)
  `\\uni\home\projects\helilab\fixation_identifier.m`
- gmt_serial_converter (in-house script)
  `\\uni\home\projects\helilab\gmt_serial_converter.m`
- instrument_selection (in-house script)
  `\\uni\home\projects\helilab\instrument_selection.m`

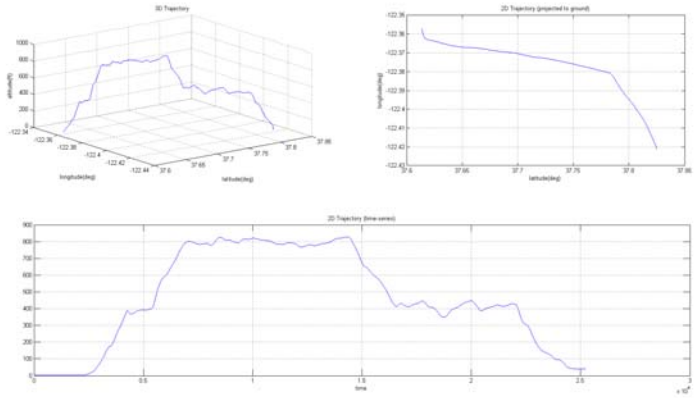


(a) Fixations during straight and level flight

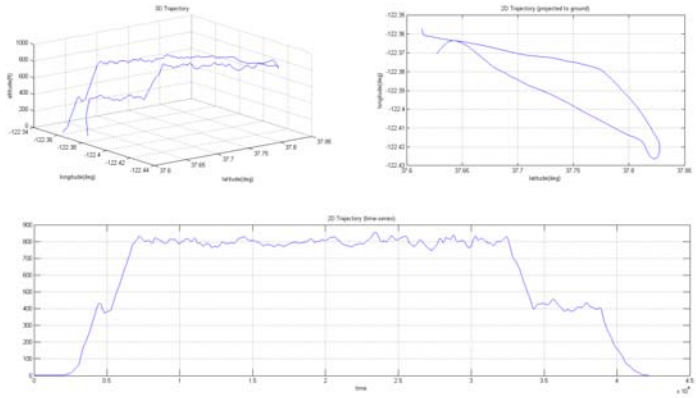


(b) Fixations during left turn

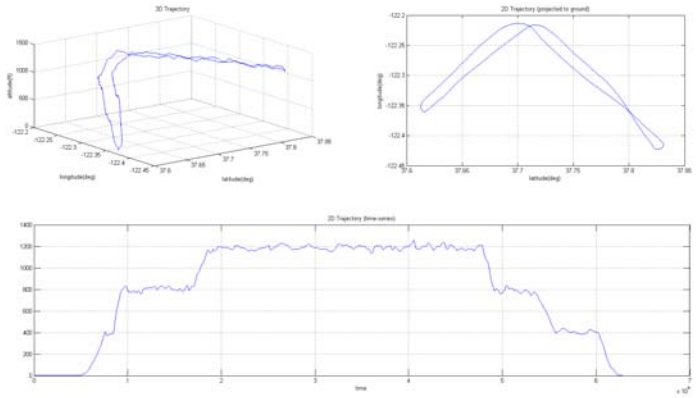
Figure 22: Fixations during flight maneuvers



(a) Trajectory of an airplane with flight plan 1



(b) Trajectory of an airplane with flight plan 2



(c) Trajectory of an airplane with flight plan 3

Figure 23: Trajectory of an airplane with multiple flight plans

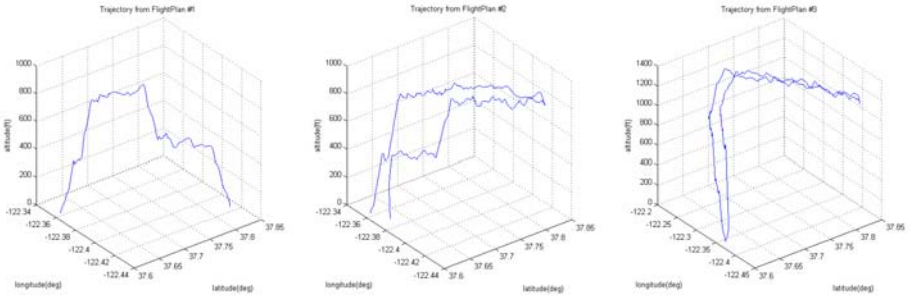


Figure 24: Trajectory of an airplane with multiple flight plans

- plot_fixations (in-house script)
  `\\uni\home\projects\helilab\plot_fixations.m`

A.3 Flight Data Analysis

- PlotTrajectory_SingleFlightPlan (in-house script)
  `\\uni\home\projects\helilab\PlotTrajectory_SingleFlightPlan.m`
- PlotTrajectory_MultipleFlightPlans (in-house script)
  `\\uni\home\projects\helilab\PlotTrajectory_MultipleFlightPlans.m`

B Appendix: Supporting Materials

B.1 Commercial Software

- MATLAB®
  `http://www.mathworks.com/products/matlab/`
- MATLAB® Database Toolbox
  `http://www.mathworks.com/products/database/`
- faceLAB™ 5
  `http://www.seeingmachines.com/product/facelab/`

B.2 Open-Source Software, Free-wares and Scripts

- MySQL™
  `http://www.mysql.com/downloads/mysql/`
- MySQL™ Connector (ODBC)
  `http://www.mysql.com/downloads/connector/odbc/`
- NTP FastTrack
  `http://www.seeingmachines.com/support/downloads/#8`

- Worldview
  http://www.seeingmachines.com/support/downloads/#9
- LogConverter
  \\uni\home\projects\helilab\logconverter.exe
- WorldModelMaker (in-house script)
  \\uni\home\projects\helilab\WorldModelMaker.xlsx
- World Model for an eye-tracking device on top (in-house script)
  \\uni\home\projects\helilab\World_Model_FL1.w3d
- World Model for an eye-tracking device on bottom (in-house script)
  \\uni\home\projects\helilab\World_Model_FL2.w3d

B.3 Manuals and Unpublished Reports

- Multiple Monitors in Flightgear: Quick and Dirty
  http://www.inkdrop.net/dave/multimon.pdf
- Guide to Use NTP FastTrack
  http://www.seeingmachines.com/downloads/NTP_FastTrack_Guide.pdf
- faceLAB User Manual
  \\uni\home\projects\helilab\faceLAB_User_Manual.pdf
- faceLAB LINK User Manual
  \\uni\home\projects\helilab\faceLAB_LINK_User_Manual.pdf

B.4 Additional Documentations and Files

- Bo105 Log (in-house batch file as a supplement)
  \\uni\home\projects\helilab\bo105_log.bat
- Log Protocol (in-house XML file as a supplement)
  \\uni\home\projects\helilab\log_protocol.xml
- General Information of Three Locations (in-house document as a supplement)
  \\uni\home\projects\helilab\General_Information_of_Three_Locations.docx
- Time-Altitude Chart of Flight Plans (in-house document as a supplement)
  \\uni\home\projects\helilab\Time-Altitude_Chart_of_Flight_Plans.pdf
- Basic Commands to Record Annotation IDs(in-house document as a supplement)
  \\uni\home\projects\helilab\Basic_Commands_to_Record_Annotation_IDs.docx
- Key Combinations Correspond to Task (in-house document as a supplement)
  \\uni\home\projects\helilab\Key_Combinations_Correspond_to_Task.docx
- New 3-Column Flight Phases (in-house document as a supplement)
  \\uni\home\projects\helilab\New_3-Column_Flight_Phases.docx
- (Experimental Purpose) Flight Plan (in-house document as a supplement)
  \\uni\home\projects\helilab\Flight_Plan_(Experimental_Purpose).docx
- Plane_Calibration (in-house image file as a supplement)
  \\uni\home\projects\helilab\Plane_Calibration.png
- Control_Calibration (in-house image file as a supplement)
  \\uni\home\projects\helilab\Control_Calibration.png

Acknowledgements
This research was supported by the myCopter project which is funded by the European Commission under the 7th Framework Program. Heinrich H. Bütholff was also supported by the WCU(World Class University) program through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (R31-10008). The authors thank Jakob Kolb, Daniel Riedel, Hans-Joachim Bieg, Joost Venrooij, Frank M. Nieuwenhuizen and Lewis L. Chuang for their assistance in setting up the flight simulator and discussions.

References

[1] F. M. Nieuwenhuizen, M. Jump, P. Perfect, M. D. White, G. D. Padfield, D. Floreano, F. Schill, J.-C. Zufferey, P. Fua, S. Bouabdallah, R. Siegwart, S. Meyer, J. Schippl, M. Decker, B. Gursky, M. Höfinger, and H. H. Bülthoff. myCopter: Enabling technologies for personal aerial transportation systems. In *Proceeding of the 3rd International HELI World Conference 2011*, pages 1–8, 11 2011.

[2] A.R. Perry. The flightgear flight simulator. In *2004 USENIX Annual Technical Conference, Boston, MA*, 2004.

[3] D.D. Salvucci and J.H. Goldberg. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 symposium on Eye tracking research & applications*, pages 71–78. ACM, 2000.