2007

Development of a synthetic vision system for general aviation

Jason Christopher Wenger
University of Iowa

DEVELOPMENT OF A SYNTHETIC VISION SYSTEM FOR GENERAL AVIATION

by

Jason Christopher Wenger

A thesis submitted in partial fulfillment of the
requirements for the Master of Science degree
in Electrical and Computer Engineering
in the Graduate College of
The University of Iowa

July 2007

Thesis Supervisor: Associate Professor Thomas Schnell

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

_____

MASTER'S THESIS

_____

This is to certify that the Master's thesis of

Jason Christopher Wenger

has been approved by the Examining Committee for the thesis requirement for the Master of Science degree in Electrical and Computer Engineering at the July 2007 graduation.

Thesis Committee: _____
Thomas Schnell, Thesis Supervisor

_____
Jon Kuhl

_____
John Robinson

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Adapted from
Marcus Tullius Cicero
The Purposes of Good and Evil

ACKNOWLEDGEMENTS

The Synthetic Flight Bag project would not have been possible without the collaboration of many individuals at NASA and the Operator Performance Laboratories of the University of Iowa. I would like to thank Monica Hughes and Lou Glaab, NASA Langley Research Center (LaRC), for friendship, collaboration, and leadership throughout the phases of this project; Tom Schnell for managing and overseeing the project, and for his advice on this thesis; Mike Keller for requirements capture and data analysis; Andy Pelzer, Royce Fullerton, and Nick Lorch for software development and hardware assembly; Jim Wagner, and Justin Regenwether for software development; Carl Richey, Nathan Kleffman and Jeff Konz for hardware assembly; Forrest Holly for flight test planning, Forrest Holly, Dale Yoder, and Tom Schnell, for work as safety pilots during flight test, Marvin Roshek for flight test aircraft maintenance; Kyle Ellis for flight test engineering; and Nick Lorch for flight test analysis.

ABSTRACT

Synthetic Vision is an aviation technology that uses databases and position estimation to establish a view of the database which provides an intuitive view that corresponds to the features of the outside world. The Synthetic Flight Bag is a low cost, portable system which implements synthetic vision, moving map, and route planning in a single software and hardware package.

Human factors analysis was performed to identify appropriate functional requirements for the development of the system. Preliminary simulator testing identified requirements on screen size and mounting location with a mind to the cramped general aviation cockpit. Hardware survey identified appropriate computing platform targets.

Hardware selected was a compact motherboard intended for embedded systems applications and graphics support. It was packaged into a custom-built avionics case, along with supporting power and I/O hardware. An LCD display with touch screen was designed and built, and represents the smallest, yet highest resolution display commercially available at this time. Software development led to a complete system with a primary flight display, multi function display, vertical profile display, and a menu and information system allowing for flight plan editing.

A flight test aircraft was instrumented and outfitted with the Synthetic Flight Bag system. A ground simulator was also created for the purpose of training prior to flight test. VFR and IFR pilots participated in the study, and were evaluated on flight technical errors, workload, and eye movement.

A flight test was performed, and results indicated that while the Synthetic Flight Bag system improves terrain awareness, it is not in its tested version a complete solution to the problem. The system was found to significantly improve the accuracy of flight, but was also found to increase workload in pilots not yet familiar with its operation. Several future improvements were identified, but the system as designed meets the project needs.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

Synthetic Vision is an aviation technology that uses databases and position estimation to establish a view of the database which provides an intuitive view that corresponds to the features of the outside world. The Synthetic Flight Bag is a low cost, portable system which implements synthetic vision, moving map, and route planning in a single software and hardware package. An illustration of the screen display of the Synthetic Flight bag can be seen in Figure 1. This paper describes the design of the Synthetic Flight Bag.

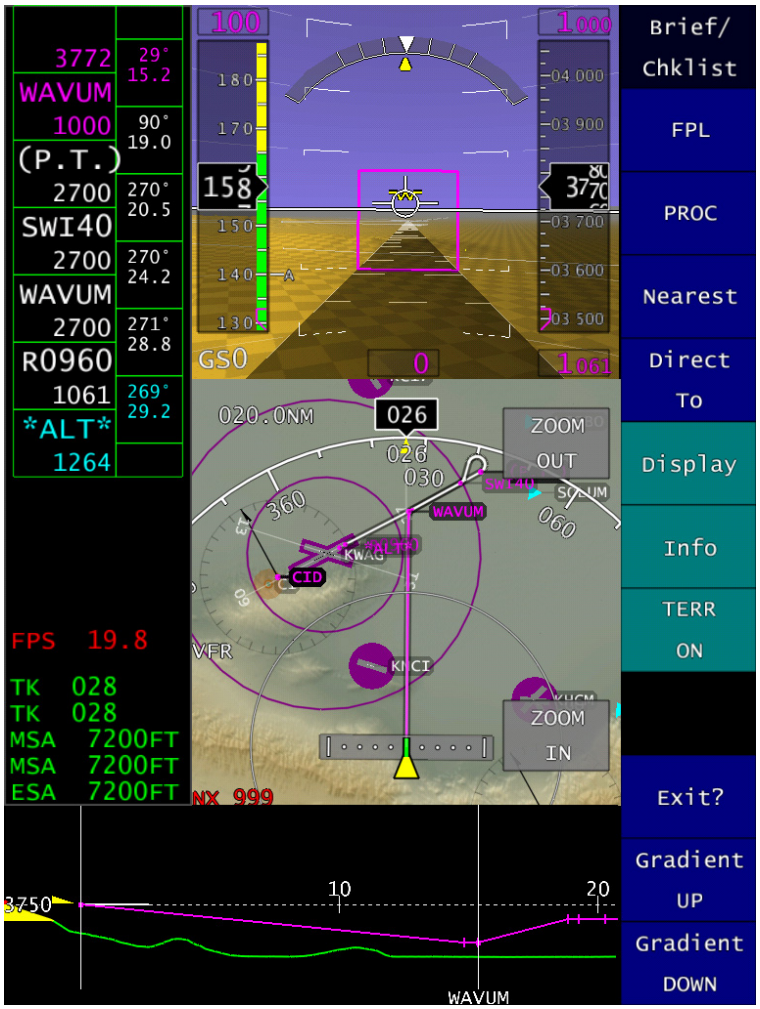


Figure 1 The main screen of the Synthetic Flight Bag software.

Statement Of The Problem

In the light of continuing improvements in safety in aviation, controlled flight into terrain (CFIT) remains as one of the largest causes of accidents in both general aviation and commercial aircraft. The primary cause of CFIT accidents is a lack of awareness of surrounding terrain, while operating in poor visibility due to weather or night. In these types of operations in today's cockpit, pilots must rely on paper charts in order to understand their relationship to nearby threatening terrain. Pilots must estimate their position on the charts by finding their location relative to nearby radio navigation aids, and mentally projecting that relative location onto the chart. This adds a heavy workload to a pilot, especially during climb to cruise altitudes and descent to landing, which are already substantially busy times in the cockpit. Moving map and glass cockpit displays represent an improvement due to the simpler presentation of the aircraft's position relative to navigational aids, but also require additional attention diverted away from primary flight instruments.

A Synthetic Vision System (SVS) is a system that combines a terrain database, a position sensor, a computing platform, and a display, to present a computer-generated perspective view of the database, as viewed from the position of the pilot in the cockpit. This display is traditionally placed on the attitude indicator portion of a glass cockpit display, and allows the pilot a very natural and intuitive view of the terrain features ahead. It is believed that the addition of an SVS display to general aviation aircraft will reap safety benefits by reducing the number of CFIT accidents. An example of the synthetic view of terrain that can be depicted may be seen in Figure 2 on page 3.

NASA, in the Aviation Safety Program (AvSP), conducted research on Synthetic Vision Systems (SVS). As a continuation of that research, the Capstone program resulted in the Chelton system entering the marketplace as a commercial product. As a certified instrument, this system may be installed in aircraft and used as a primary instrument. However, the costs of certification result in the Chelton system being most likely too

expensive for the general aviation community to afford. Because of this, there is a need for a lower-cost system to bridge the gap and provide some SVS related safety benefits in a non-certified application.

As a non-certified instrument, this lower cost system would not be able to be permanently installed in an aircraft. In addition, pilots would not be able to use the instrument as their primary reference for flying or navigating the aircraft. However, in the same way that a portable GPS can be temporarily installed and provide for significant workload reduction and safety benefits, it is believed that a portable SVS system could

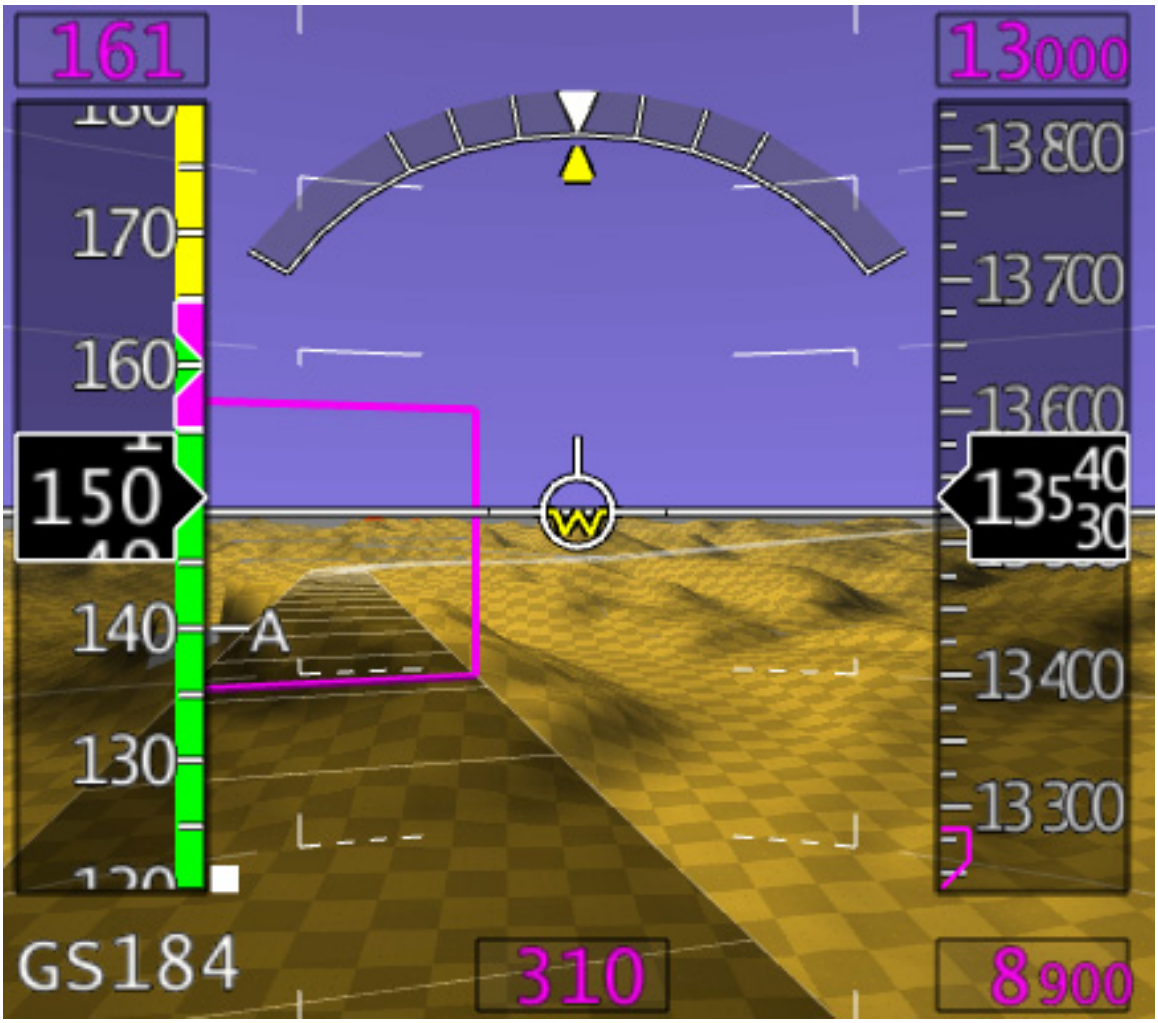


Figure 2 A depiction of a synthetic vision display

have similar advantages to a pilot flying in low-visibility conditions.

Solution Approach

A NASA funded research program at the University of Iowa's Operator Performance Laboratory was very specifically geared towards creating a fully functioning system that could be brought to the market as a product. The development work presented in this thesis was conducted as an integral part of the NASA funded effort.

Prototyping of Synthetic Vision has been done extensively in the past by a number of researchers, always in a limited geographical area, resulting in prototype systems that are usually designed for approach and landing in a specific terminal environment. All of these systems have been hand tailored to a specific airport, and have not dealt in detail with route planning. The approach chosen by OPL and reflected in this thesis was to develop a fully self-contained and ruggedized system that works in all geographical areas and derives its guidance, terrain, obstacle, and symbology information from official government databases rather than hand-tailored scenarios.

The proposed solution to building such a system was to rely heavily on Commercial Off-The-Shelf (COTS) hardware to build a non-certified system. As a non-certified system, use in an aircraft would only be possible either via a permanent installation approved on an aircraft-by-aircraft basis in a difficult field approval process, or with no necessary approval process, if the system is portable. This drove a very strong requirement that the system would be portable, able to be installed in a new aircraft within a few minutes, and able to be removed as quickly. As a portable system, it would not be possible to integrate with the existing aircraft systems in any meaningful way; therefore, it would be necessary for this portable system to include its own navigation sensors.

Finally, in order for the system to be in reach of the average general aviation pilot, the whole system was designed to have a price in the $5,000 - $10,000 range. Two

variations of the system were produced. The first variation was an advisory only display, intended for production as an add-on system to an existing aircraft. The other was to demonstrate a more integrated system, perhaps with multiple displays, approaching a complete glass cockpit flight deck. This 'flight critical' setup was intended only as an experimental display, as the costs associated with producing a certified version of such a display would be far too high to be applicable to the intended purpose of the display. These two display format requirements drove many of the architectural and inter-process communication requirements of the system described herein.

Background And Literature

Lemos & Schnell (2003) studied the effects of terrain data resolution, texture, and lighting on pilot's ability to identify differences in views of terrain, and their ability to navigate down the center of a valley. They found that a terrain data resolution of three or six arc seconds is sufficient, and that Gouraud shaded images with a checkerboard or elevation-colored texture produced lower workload.

Lemos, et al. (2002) performed a comparison of various glass cockpit displays. Features and display resolutions of available flight decks were examined and compared.

Schnell, et al (2003) examined field of view and pixel resolution requirements for synthetic vision displays. 105 pixels per inch was found to be a sufficient resolution at the screen size studied. Field of view was found to be optimal at 60-90 degrees for initial approach and navigation, and from 30-60 for final approach.

CONCEPTUAL DESIGN

Determination of pilot needs in a portable navigation and synthetic vision system was an important part of the design process. To establish these needs, user requirements were captured via a focus group and online survey. Display size requirements were established with a fixed base pilot study. An extensive hardware search was performed, to find candidate hardware platforms for the implementation of the Synthetic Flight Bag.

Requirements Capture

A focus group was held to collect requirements for both VFR and IFR operations. Following the creation of this list of requirements, an online survey was created. 327 pilots completed this survey. Each pilot rated the captured requirements for importance and frequency of use. The results of the survey are included in Table 1 on page 7. The features rated most important were ruggedness and lack of cable clutter, availability of a direct-to function, inclusion of a full aviation navigation database, ease of updates of the aviation database, and integration into the overall instrument panel. The features rated most likely to be frequently used were the direct-to function, a full aviation navigation database, direct-to functionality specific to the nearest airport, and overlay approaches on a moving map display.

Two studies were performed in a fixed base simulator, the first examining display size and resolution, the second examining field of view and display size. Each study was performed in the OPL General Aviation Work Station 2 (GAWSTOO) simulator. This consisted of an Elite Pro Panel yoke, a projected outside visual system, and a back-projected pair of head-down displays. These back-projected displays could be adjusted in size by adjusting the position and arrangement of the projectors, combined with lens zoom. Brightness was controlled with a neutral density filter. This setup allowed display size and resolution to be varied independently. Another independent variable was whether the display was used as a primary reference or a supplemental display.

Table 1 Rankings of importance and frequency of use of captured requirements.

Feature	Importance						Frequency of Use					
	Very Important	Important	Somewhat Important	Not Important	Don't Know	Other	Very Often	Often	Sometimes	Never	Don't Know	Other
Direct to Fix	220	68	25	6	2	0	251	56	6	1	1	2
Direct to Nearest Airport	65	82	157	14	1	2	208	83	25	0	0	2
METAR Decoding	78	52	58	102	19	5	128	79	59	24	3	15
Wide Area Augmentation System (WAAS)	52	85	58	102	21	8	117	88	37	6	0	46
Terrain Awareness and Warning System (TAWS)	48	48	62	135	21	0	145	86	32	5	0	33
Altitude Alerter	53	71	78	102	10	4	118	98	67	9	1	17
Integration with Auxillary Systems	68	64	48	100	27	7	154	88	28	3	1	26
Obstacle Database	107	82	76	48	2	4	176	79	43	5	1	6
Sectional / Moving Map	105	90	57	57	4	5	174	82	36	5	1	27
Landmarks on Moving Map Display	108	115	82	16	0	1	141	113	55	6	1	1
Complete Jeppesen Database	172	75	53	16	5	0	225	71	16	1	0	6
Procedures for Departures/Arrivals for Airports	82	92	104	33	6	2	176	92	33	5	1	8
Overlay Appr/Dep on Moving Map Display	98	83	58	67	13	0	205	60	23	4	0	16
Edit/Create Active Flight Plan	131	84	65	33	2	2	145	91	57	10	2	8
Insert Waypoint Into Flight Plan	111	77	110	19	1	0	164	99	46	4	0	4
Creation of User Defined Waypoints	53	65	166	33	1	0	110	109	79	14	2	3
Voice Prompt Guidance	15	24	53	170	43	10	39	67	88	39	7	59
Smart Keys	28	57	90	102	33	5	62	103	79	15	6	34
USB Support	129	107	53	14	16	0						
Serial Support	32	54	104	101	26	1						
Long Battery Life / DC Connection	197	73	25	12	7	6						
Ruggedness; No Cable Clutter	208	89	19	1	0	1						
Yoke Mountable	62	95	91	57	8	4						
Integration into Instrument Panel	151	87	58	14	16	0						
Loadable data cards for updates, expansions	184	98	27	4	3	3						

Pilots were measured on vertical and horizontal flight errors, track angle and flight path angle error, speed error, touchdown point dispersion, control inputs analysis, control wheel zero crossings, secondary task performance, and eye fixation data. Each pilot flew a terrain following task, and an approach and landing task. Results indicated that display size was not a significant driver of any measures, provided that resolution was sufficient. This indicated to us the feasibility of developing on a significantly smaller than usual display. Further details of the results may be found in "Pilot Performance as a Function of Display Resolution and Field of View in Simulated Flight Using Synthetic Vision Systems." (Keller et al, 2003)

Hardware Search

An extensive hardware search was performed. Initial questions were raised as to whether a PDA or cell phone solution might be possible. It was found that there are no practical ways to do this, given the extensive hardware requirements of synthetic vision. There simply is not enough capability in a PDA to perform the tasks needed.

With those platforms eliminated, attention turned to standard PCs as a computing platform. Of those, two major contenders existed. The first would be to use a laptop in a docking station. However, this was found unsuitable from the standpoint of ruggedness and lack of cable clutter. A second option is to use an embedded platform computer with adequate graphics, coupled to a battery included with the system. This solution was found to be acceptable.

Many AHRS and position systems were evaluated. Many outstanding AHRS solutions, such as those manufactured by Crossbow, were discounted entirely, due to the high cost of the systems. The Cloud Cap Technology Piccolo Plus was investigated and found to be good, but still too expensive for the final system. The Rotomotion, LLC AHPRS is a less expensive system, as is the PCAvionics eGyro-XP. Both of these were investigated and found to be unsuitable for use in flight. The Rotomotion AHRS was

found to operate correctly on the table in the lab, but is unable to accurately estimate attitude when strapped down to an accelerated platform such as an aircraft, often indicating significantly flawed estimates of pitch, roll, or yaw. Even worse, even moderate maneuvering was found to have the potential to entirely upset the system, causing it to indicate attitude as though perpetually tumbling in all three axes. Meanwhile, the eGyro-XP, which has no external aiding, was found to have very poor performance over time, often drifting far from the correct attitude, especially during repeated turns, such as when holding. Finally, the Dynon EFIS-D10A was investigated and found to be an appropriate solution for the purposes of development.

Various displays were evaluated, and a 8" 1024x768 display was quickly found and selected. Other displays evaluated were a 7" 800x480 pixel widescreen display, and a larger 1024x768 VGA display.

Several power systems were prototyped, evolving eventually to the power system described later. The components of the final hardware system are the same as initially prototyped, but development lead to simpler packaging of the initially chosen components.

Format Development

Format development was iterative, and changed as various physical displays were worked on. It was decided from early on in the project to use a menu system for control of the display, focusing nearly all of the input into one small portion of the display. Consideration was given to input of flight plans, and a telephone-keypad system was selected as most suited for use on the display. Initially, a hardware keypad was envisioned, but use of a touch screen display allowed reuse of screen space for that purpose, leading to a simpler design. The general arrangement of the display was found quickly, but continued to evolve throughout the course of the project.

HARDWARE

Basic requirements found in the conceptual design phase drove specific constraints into the hardware solution chosen. First, it was necessary that the system be rugged enough to be used in an aircraft environment, yet also portable, able to be installed or removed from an aircraft in minutes, without permanent mounting. Secondly, it must operate on the power sources available on the aircraft. One commonly available power source on an aircraft is an equivalent to an automobile cigarette lighter, providing 12-14.5V DC. Some aircraft, however, operate on 28V electrical systems, and the system should be able to operate in that environment as well. The hardware solution must be able to provide enough graphics performance to produce an acceptable synthetic vision image. Finally, the system must be safe for use in an aircraft, especially with regard to fire safety.

Computing

The primary computing hardware for the Synthetic Flight Bag is a Commell LV-671, shown in Figure 3 on page 11. The Commell LV-671 is an industrial computer motherboard in the Mini-ITX form factor (170 mm x 170 mm), and is powered by an Intel Pentium M processor.

It is unique among most industrial motherboards in that it has a proprietary Mini-AGP connector, allowing for the installation of a daughterboard containing an ATI M10 (Mobility Radeon 9600) OpenGL accelerator. In addition, the Commell LV-671 needs only 12V DC as a power supply, compared to most industrial computers, which require a full ATX power supply. This allows for a significant reduction in wiring harness complexity and weight.

The Commell LV-671 offers two serial ports, six USB ports, a single PCI slot, and built-in onboard sound and Ethernet. It accepts a single DDR333 SDRAM module. With a socket that accepts up to a 4-gigabyte CompactFlash card, which is seen as a

standard hard drive, there is an option to use a built-in solid-state disk, thus allowing for higher reliability than a hard drive storage solution.

Should a hard drive be used, another uncommon feature is the ability to connect to either desktop or notebook hard drives. If a notebook hard drive is used, it is connected by a ribbon cable that includes built-in power and ground wires, again reducing the complexity of the cabling needed.

Figure 3 The Commell LV-671.

Note: Size is 170x170 mm. The motherboard supports a Pentium M processor, DDR333 SDRAM module, an onboard graphics accelerator, and can operate on 12V DC power.

Graphics

Equally important to the computing requirements of the synthetic vision application is the performance of the graphics accelerator. In most desktop systems, this would be added to the system in the form of an expansion card that would be plugged into an AGP or PCI-Express slot. Due to the focus on reducing the size and weight of the computing hardware, this was deemed an unsuitable solution. Instead, the Commell MA-ATI video card is used, seen in Figure 4.

This card contains the same hardware as a standard ATI Mobility Radeon 9600, but is packaged into a special form factor that can be installed only into the proprietary Mini-AGP connector found on the Commell LV-671. In this case, the motherboard and video card form a paired solution, each working only with the other.

The Commell MA-ATI video card can produce outputs of VGA, DVI, composite video, or LVDS. LVDS, (Low-Voltage Differential Signaling) is a lower-level interface than the more well-known VGA or DVI interfaces. Traditional desktop LCD monitors accept VGA or DVI signals at varying resolutions, color depths, and refresh rates,

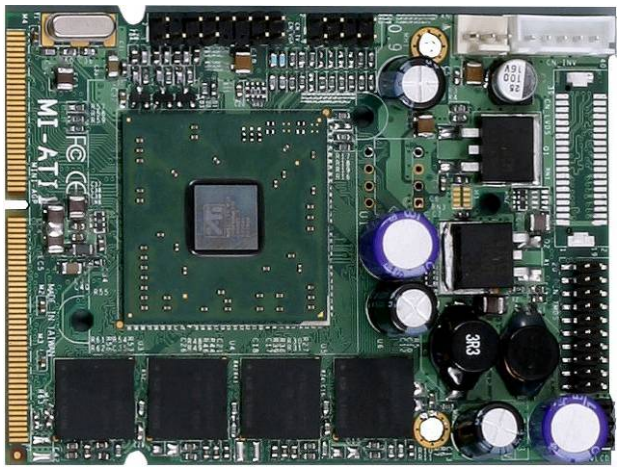


Figure 4 The Commell MA-ATI card.

Note: This card contains an ATI Mobility Radeon 9600 graphics processor and is a special-purpose part which can be used only with the Commell LV-671.

however the actual LCD panel will work with only one specified resolution, color depth, and refresh rate. Because of this limitation, a processing card in the display will condition the VGA or DVI signal, converting it into the specific resolution, color depth, and refresh rate that the LCD panel will accept, and send those conditioned signals via LVDS to the LCD panel. The Commell MA-ATI is uncommon and unique among video cards in its ability to output LVDS signals directly, thus avoiding the need for a DVI converter board in the display head, allowing for lower weight and a thinner display.

Sensors

The final component needed for the Synthetic Flight Bag is a GPS receiver board. The board selected is the uBlox RCB-LJ, using the uBlox ANTARIS GPS engine. The board contains an MCX antenna connector, will provide power to antennas that require it, and communicates via a pair of TTL serial ports. Standard RS-232 serial ports communicate using signaling at ±12V. A TTL serial port uses the same signal timing and data format, but operates at 0V-5V DC. It is necessary to add a level-shifter to convert to standard RS-232 voltages. The RCB-LJ is shown in Figure 5.

An optional, but recommended addition is the Dynon Avionics EFIS-D10A, an inexpensive attitude, airspeed, and altitude indicator, designed for experimental category

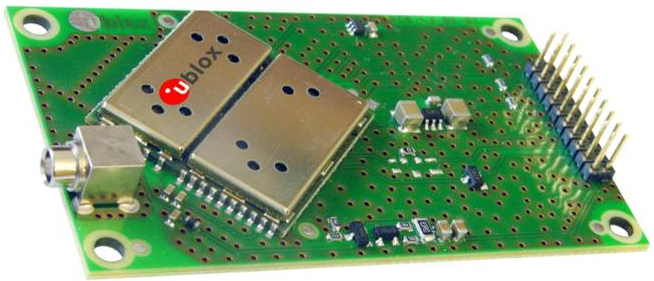


Figure 5 The uBlox RCB-LJ

Note: MCX-type antenna connector is visible at lower left, pin header with power and serial ports at upper right.

Figure 6 The Dynon Avionics EFIS-D10A

aircraft, seen in Figure 6. While the Synthetic Flight Bag system can be used, with reduced capability, without this information, its availability significantly improves the usefulness of the overall system. The EFIS-D10A requires permanent mounting to the aircraft panel, and thus is not suitable for a temporary installation. In these cases, it would be necessary to support an alternate attitude source that could be installed in a temporary manner.

Power

An important consideration in the development of the Synthetic Flight bag was that of power. It is necessary for the system to be able to be powered off a cigarette lighter outlet, which depending on the aircraft, may be a 12 or 24 volt power source. It is important for the system to continue operating if the engine is not running and aircraft battery power is not available, to allow the day's flight plan to be entered before engine start. This also allows the system to be used in emergencies if the aircraft loses electrical power. Both of those situations necessitate an internal battery. However, as many aircraft fly only infrequently, often with spans of several months between flights, it is

important that the battery be completely disconnected from any components that may draw power when the system is not running, to allow the battery to maintain its charge during long gaps between flights. To this end, a power system was developed with these needs in mind. This power system can be seen in Figure 7 and Figure 8 on page 16. The creation of this power system involved the integration of several power components.

The battery chosen is a standard Lead-Acid battery, the Panasonic UP-RW1245P1. While the Lithium-ion batteries that are used in most laptops provide far more power given the same weight, they require far more complicated charging

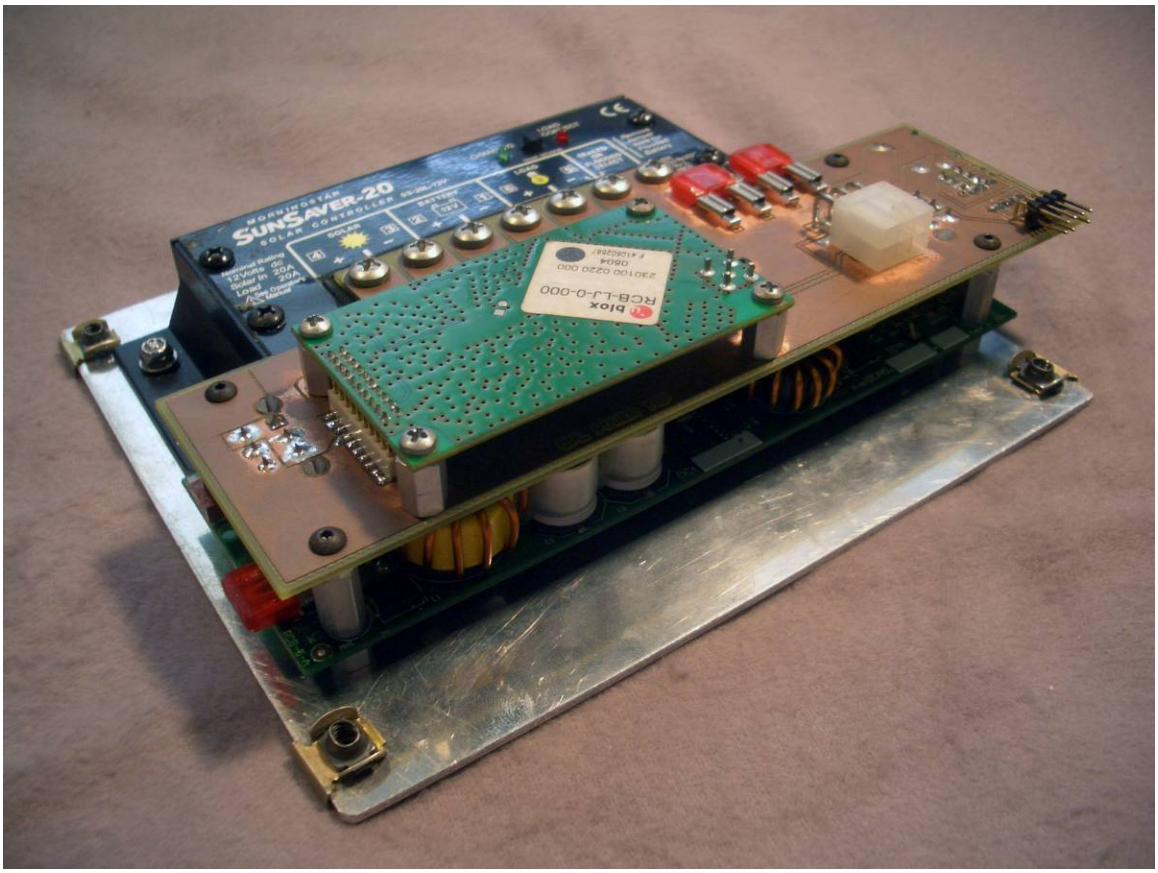


Figure 7 The combined power system.

Note: At back can be seen the Sunsaver, a battery charge controller. The copper colored board is the power distribution board. Mounted below that board is the Opus Solutions power conditioner. The uBlox RCB-LJ is also mounted to this board for ease of packaging.

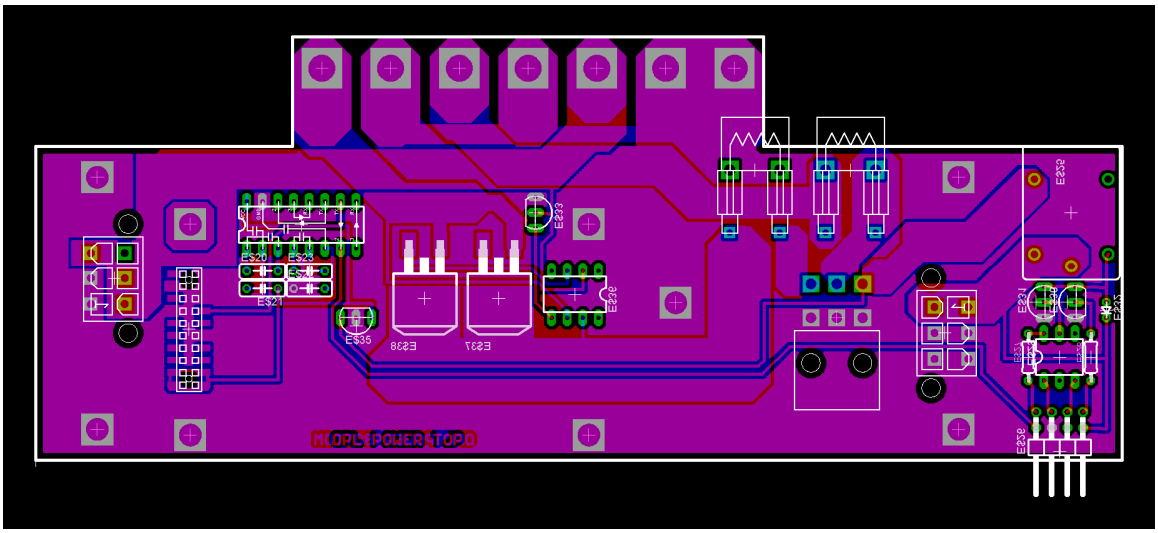


Figure 8 The layout of the power conditioning board.

Note: At top, connection pads to the Sunsaver. At right, switching circuitry to disconnect the battery when not in use. At center, FET switches which disconnect the battery power from the aircraft when the aircraft power is at a lower voltage than the internal battery. At left, connection points for the uBlox RCB-LJ.

equipment and significant safety measures. Recent recalls of Li-ion laptop batteries manufactured by Sony, due to manufacturing defects that rendered them likely to catch fire or explode while in use illustrate the advantage of using the safer Lead-Acid batteries chosen for this system.

The battery voltage and charging is managed by a Morningstar Sunsaver SS-20, a COTS battery charge controller normally used for small solar-power systems.

Finally, the Opus Solutions DCA5.080.12V is necessary to regulate and condition the battery-supplied voltage, which may vary from 10 to 14 volts, to a constant 12 volts, as needed by the display and motherboard. The power capacity of the board is 80 watts, enough to provide power to all components of the system with the exception of the Dynon D-10A, which is connected directly to aircraft power and has its own power conditioning and standby battery.

Finally, it was necessary to build an appropriate power-distribution system to connect the battery, fuses, charge controller, power feed, power conditioner, and the various portions of the system that need power. Initial efforts in this regard consisted of a wiring harness that was installed in the main computer box. This first prototype had two circuit breakers mounted on the face of the box, which individually protected the cigarette lighter plug and the battery from damage or fire due to excessive current. Construction of this wiring harness, however, was found to be one of the most time-consuming tasks in the construction of the system. Later revisions replaced this complicated wiring harness with a printed circuit board with connectors that allowed all the parts to be snapped together and then bolted in place. Wiring was used only to connect to the battery terminals and the motherboard, saving significantly on difficulty of construction, and providing better airflow for cooling. In the later boxes, the circuit breakers were replaced with internal fuses, saving weight and cost.

Once the system was built and testing had begun in the aircraft, one unforeseen problem was discovered. When the aircraft engine was at idle, the fully charged battery in the processing box would often provide a higher voltage than the alternator. In these cases, power would flow out of the box and into the aircraft. In essence, the aircraft battery and the processing box's standby battery would work in unison to attempt to power the aircraft. This would cause an audible interference in the headsets and strained the circuitry in the computer box with unexpected power load from the aircraft electrical system. In addition, if the aircraft power was shut off, the box's standby battery would attempt to power the entire aircraft's avionics systems. This would cause the battery circuit breaker to trip, if enough equipment on the aircraft was left on. In other cases, it would simply lead to the discharge of the battery. Both of these cases were undesirable, and so it was necessary to add circuitry to ensure that the connection to the aircraft would be disconnected in situations where the aircraft power was at a lower voltage than the internal battery. Adding this circuitry solved both of those problems.

Main Box Packaging

Given the strong desire for ruggedness seen in the market survey, a decision was made to use standard Churchill 3/8 ATR (Air Transport Radio) avionics enclosures to house the main processing hardware. An example of the final assembled box can be seen in Figure 9. These boxes come in a standard set of sizes and allow for mounting in standardized trays. This proved highly useful as the aircraft was instrumented, allowing the boxes to be securely mounted, yet also to be quickly removed should service or software changes be needed.

Figure 9 The assembled Synthetic Flight Bag Computer.

Note: The case is a 3/8 ATR avionics case. Visible at left are the four connectors to the case. The small gold plug is the antenna connector. The large plug beside the antenna connector is the display connector. At bottom, partially obscured behind the handle, are power and interface connectors.

Components were mounted on metal standoffs using countersunk holes in the sides of the box. The battery was secured with a metal strap, to the rear of the box, allowing it to be removed separately. A wiring harness was assembled to connect the computing components to the power components and front panel connectors. Figure 10 provides a closer view of the motherboard and its connection to the faceplate and other external connectors.

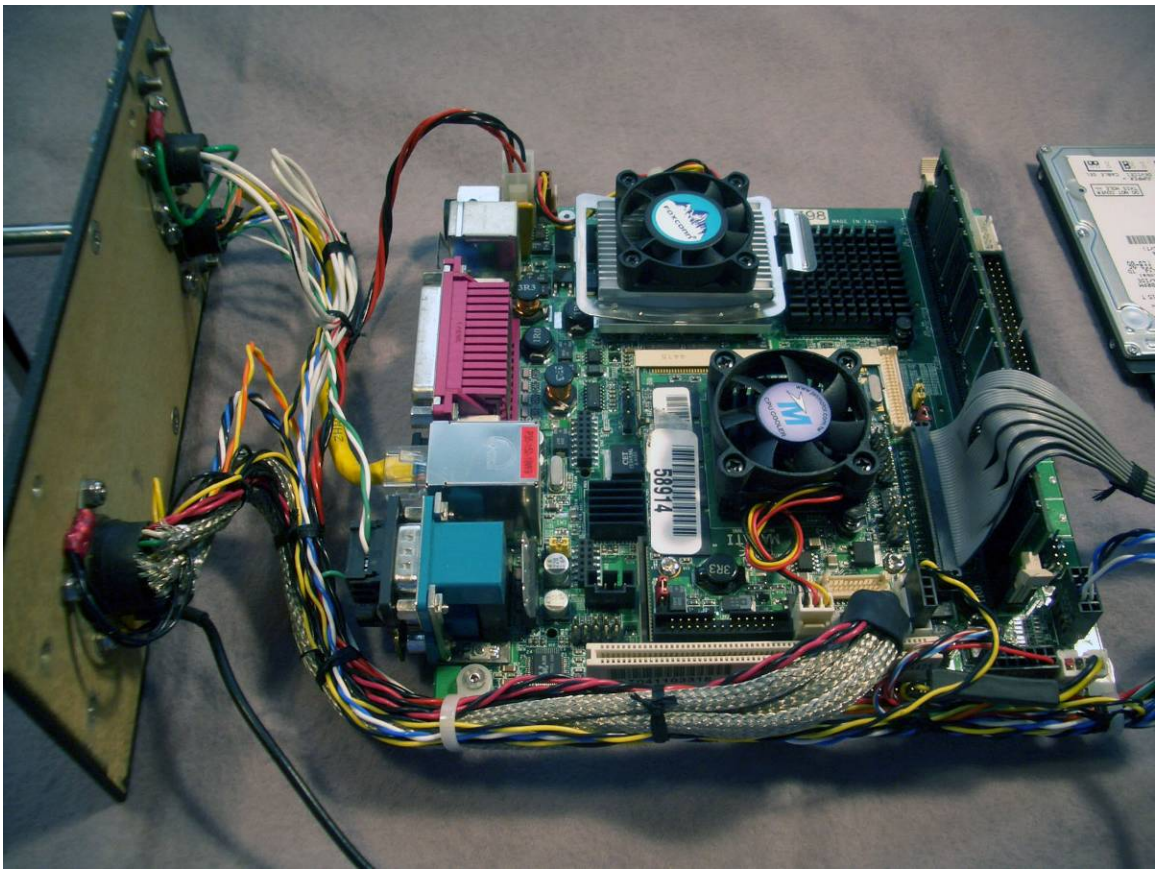


Figure 10 A closer view of the LV-671 and front plate

Note: Note the wiring harness connecting the LV-671 and the front plate. The display connector, seen at lower left, connects LVDS video connections, USB connectors, and a power switch to the motherboard. The power connector, at the top of the image with the green wire, connects power and ground through the harness to the power system. The external interface connector, beside the power connector, attaches to the Ethernet and serial ports on the motherboard.

The full contents of the main box can be seen in Figure 11. The computer board with graphics card installed is mounted on standoffs to the case. The power distribution system is installed on the opposite side of the case, attached to a mounting plate. The battery is attached to the back plate with its mounting strap.

Connectors chosen were Amphenol Circular Bayonet Lock connectors, MIL-C-26482 type, a connector type commonly used in avionics, meeting military specifications.

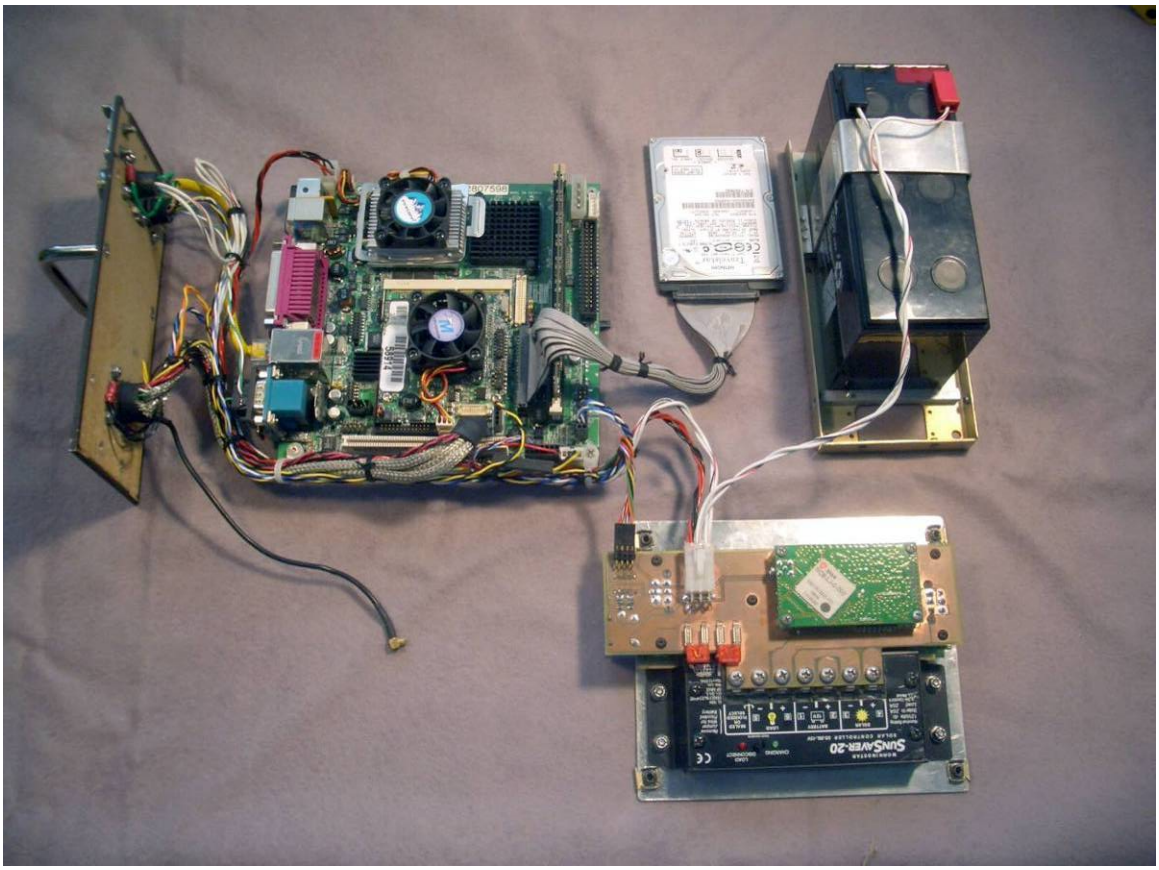


Figure 11 The contents of the Synthetic Flight Bag Computer.

Note: At left is the face plate of the 3/8 ATR avionics case with the external connectors installed. To the right of that is the Commell LV-671 and MA-ATI video card. To the right of the motherboard is a standard laptop hard drive. At far right is the battery. Visible at bottom is the power system, consisting of the Sunsaver, power distribution board, Opus Solutions power conditioning board, as well as the uBlox RCB-LJ GPS board. The black cable with a gold end which connects to the faceplate is the GPS antenna cable, and plugs into the RCB-LJ GPS board, but cannot be connected with the box disassembled as shown.

The connector is a round plug, with five keys at the edges to ensure it is connected in the proper orientation. It can be locked in place once connected with a twist-lock ring. Varying sizes and number of pins are available, and three different sizes are used on the main computer box. Figure 12 illustrates an Amphenol circular bayonet lock connector on the faceplate of the SFB main computing box.

The largest of the main box connectors has 26 pins and carries LVDS video, power, USB, and connections for a power switch to the display. A three-pin connector carries power and ground from the cigarette lighter connection. Finally, a ten pin connector provides an Ethernet connection allowing for software changes, and allowing

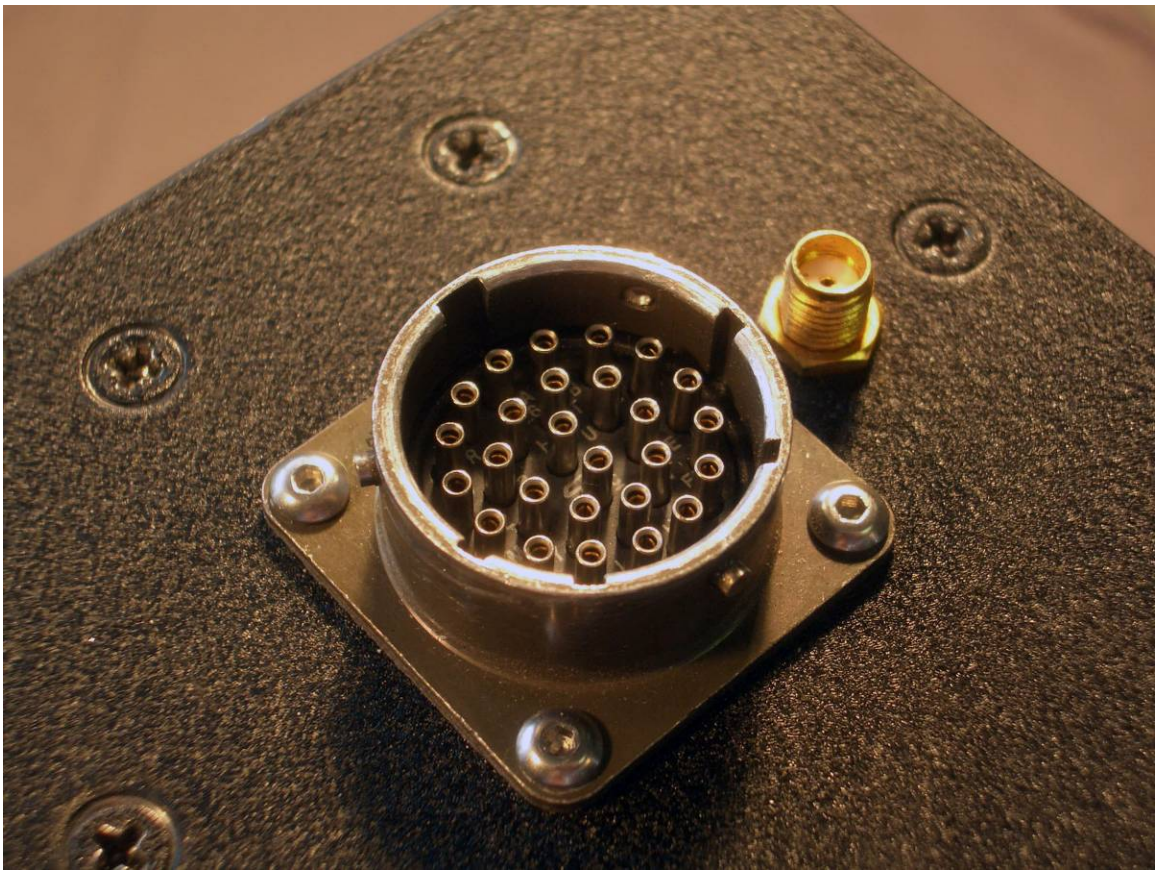


Figure 12 The display connector on the Synthetic Flight Bag computer.

Note: Amphenol Circular Bayonet Lock connectors were chosen for their robustness and pin density. Also visible, in gold, is the eternal SMA-style GPS connector

the box to be connected to simulated data sources such as a flight simulator, for bench testing. Ethernet is also used on the flight test aircraft to allow data logging, sharing of aircraft state data, and connection to the flight test aircraft's onboard NEXRAD weather receiver. Also on the ten-pin connector is a serial port, allowing connection to a Dynon EFIS D-10A, or to other attitude sources tested earlier in the development of the system. Finally, power is available on the connector, should it be needed by various attitude sources which may be connected.

Display

The display selected was the Optrex T-51639D084U-FW-A-AA, an 8.4 inch, 1024x768 pixel, daylight readable LCD screen. This display represents the smallest commercially available, daylight readable screen of that resolution at the time it was purchased. Given the difficulty in finding available space in the cockpit of general aviation aircraft, a physically small display is required. At the same time, the initial human factors studies performed in the conceptual design phase show that high resolution is also required, allowing for a clearer display, and for more information to be provided to the pilot. This display represents the best compromise in size and resolution.

The LCD screen was packaged into a custom-made aluminum display can, designed to contain the LCD and other needed components in the minimum possible space. Mounted behind the LCD is an inverter board, which converts 12V DC power from the main computing box to the voltages necessary to power the LCD display's backlight. Also needed is a small potentiometer, used to adjust the brightness of the display. A touch-screen driver board connects to the display, and interfaces via USB to the main display box. This connects through a 4-port USB hub, also mounted behind the LCD screen. This hub also connects to two USB ports, mounted at the top edge of the display, allowing for USB flash memory keys to be used, for example, for flight plan loading and transfer. The final component of the display is a power switch, which

connects to the main computer box. The display is powered whenever the computer is turned on, and shuts off automatically when the computer does. These parts and wiring are shown in Figure 13. The assembled display can be seen in Figure 14 on page 24

Input

Considerable effort was devoted to determine, test, and develop appropriate data input methods for this Synthetic Vision System. In the end, a touch-screen display was found to be the most flexible option. A touch screen is also a good choice as it does not require additional space in the cockpit for physical buttons or a keypad.

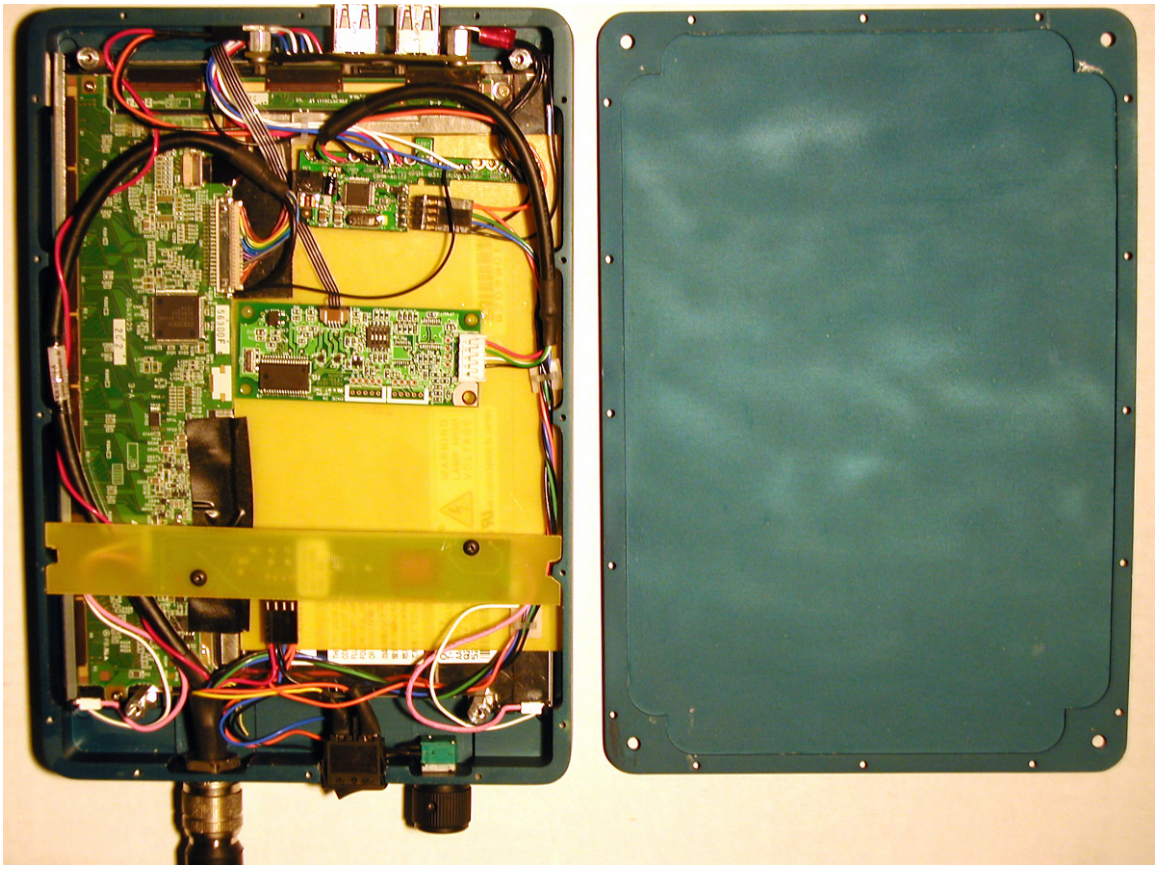


Figure 13 The inside of the display can, showing wiring harness

Note: At left, face down, is the display. At right, the back-plate is removed. The LCD is partially obscured behind the tan fiberglass sheets used for component mounting. This is an early prototype wiring harness, not designed for neatness or ease of manufacture

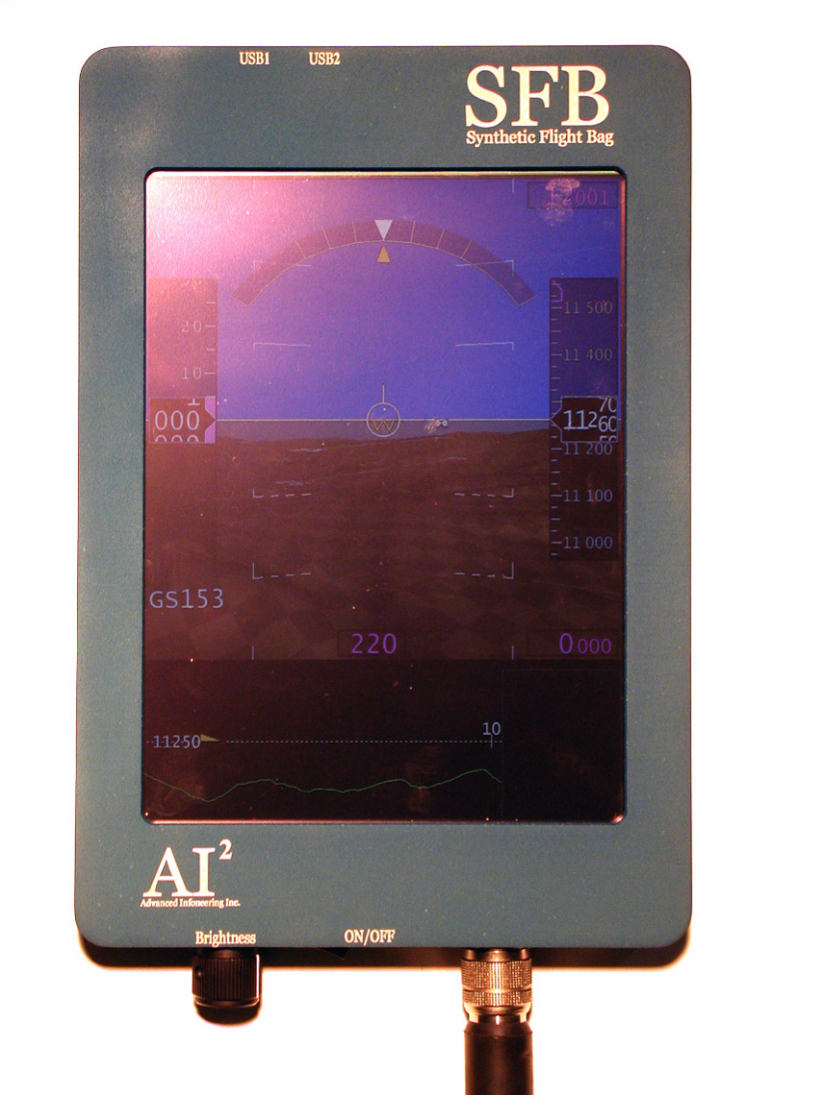


Figure 14 The Synthetic Flight Bag display.

Note: The enclosure is blue-anodized aluminum and consists of a front and back plate, held together with screws. Visible at bottom, left to right, are a display brightness dimmer knob, a power switch, and the display cable connector.

The biggest disadvantage of the touch-screen is that it somewhat reduces the daytime visibility of the LCD screen itself. Also, touching the screen in the operation of the Synthetic Flight Bag may result in fingerprints and smudging on the screen, further reducing the visibility of the display. Use of gloves when flying reduces this problem.

Many commercially available systems solve the problem of fingerprints, along with the “fat finger” problem of requiring very large buttons to be accurately pressed by a finger, by instead using a touch-screen with a stylus.

This was found to be undesirable due to two reasons. First, a stylus must either be tethered to the display, providing a noose to potentially catch on switches or entangle flight controls, or it must be free. In the latter case, it risks being dropped and lost under the seat, or some other place in the aircraft that may be difficult to access, especially in a single pilot situation. In that case, the pilot is forced either to use their finger, or improvise with some other utensil. Even worse, some displays do not react at all to any touch except the stylus itself. In this case, dropping the stylus could potentially prevent any user input or, even worse, lead to pilot spatial disorientation if a stylus needed to be picked up from the floor.

Secondly, many applications with stylus interfaces seem to have initially been developed on the desktop, where the speed and precision of a mouse drives developers to produce displays with many small buttons. In flight, confronted with even minor turbulence, it can be very difficult to use a stylus to select small controls on the display. In this case, the necessarily large buttons designed for use with the finger are in fact an advantage.

SOFTWARE

The largest part of the project was the development of the display software hosted on the Synthetic Flight Bag platform. The application was developed in C and C++. The main application is a multithreaded application, consisting of a main thread which handles the display and user input, and a number of accessory threads which manage the available data. One thread is created to load and prepare terrain data from the database. Another thread loads navigation data, such as airports, waypoints, radio navigation aids, and airspace boundaries. A third thread handles requests for specific navigation information from the main display thread. Finally, a fourth thread was created to handle data from a NEXRAD weather receiver, when used on the flight test aircraft. Communication between threads is handled through the creation of 'arenas' which store loaded data, and can be accessed in a thread-safe manner. The structure of the application is illustrated in Figure 15 on page 27.

The main thread handles a variety of software tasks related to the display, each of which is encapsulated into its own separate division in the software. Once the background threads for database access are started, the main thread enters the runtime loop, in which it services the aircraft state communication, a PFD, MFD, VPD, information window, an active and planned flight plan, and the menus. Each of these is processed and drawn independently of the others, though there are interfaces to allow each to interact with the others. For example, actions taken on the menus may affect the route, and changes in the route are communicated to the other windows. These interactions are handled through a set of standardized external interfaces. As all of these functions are in the main thread, there is no need to be concerned with thread-safety or locking in those interactions.

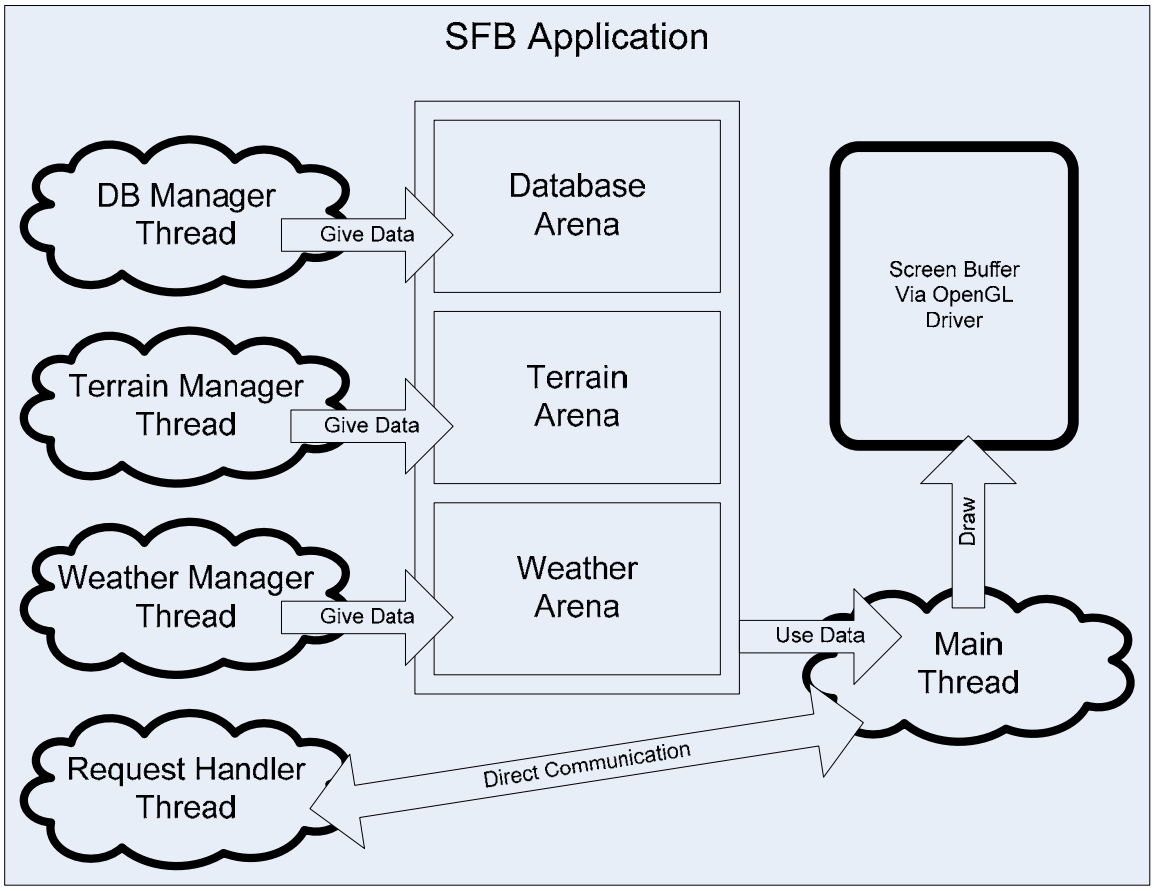


Figure 15 The structure of the SFB application

Development Environment

Software for the Synthetic Flight Bag system was developed on a single server running Red Hat Enterprise Linux Workstation release 3. The source code was stored in a CVS repository for revision control. Developers used Windows PCs and connected via SSH connections to the server, and checked out copies of the development tree into their home directories on the server. Samba shares allowed the developers to connect to the tree in their home directories via the usual Windows network drive interface. Files in the development tree were edited using Microsoft Visual Studio 6.0, but compilation was not done using this tool. Rather, all compilation, both for Linux and Windows executable

targets, was performed on the server. Compilation for Linux was done in a straightforward manner with the standard installation of GCC. Compilation for Windows was done with a Linux-hosted install of the MinGW toolchain. MinGW is a version of GCC with an independent version of the Windows header files and import libraries that allows compilation of Windows programs. By building this toolchain as a cross-compiler on the same Linux box as is used to compile and manage the code, a single build process can produce executables to run both on Linux and Windows with equal ease. Platform specific code, such as for windowing, graphics, timing, threading, and network support, was carefully constrained to a fixed set of libraries, which acted as a translation layer, allowing the rest of the project to access these functions through a common interface. No platform-specific code was allowed anywhere except in these specific libraries.

Software builds were managed through makefiles that were constructed for the entirety of the project, based on a highly modified version of the makefile system proposed in the paper, “Recursive Make Considered Harmful.” (Miller, P.A. 1998). The development tree as a whole was split into small functions which could be edited, for the most part, in isolation. Each function existed in a single subdirectory of the overall tree and was compiled into a single library. The main makefile consisted simply as a listing of the names of the directories that contained the libraries and applications, along with generic rules for building a library from source files, and an application from source and libraries. Each project had a sub-makefile which was included by the main makefile, and listed only the exceptional parameters of that library that differed from the standard generic make process, if any. This allowed for very simple makefiles in the projects; Most were simply one line, listing the name of the directory where the source files could be found. Everything else needed to build the library could be inferred from that.

To build a library, source files were built into dependency files, which automatically identified prerequisites, via the mechanism identified in section 4.14 of the GNU make manual (Stallman et al, 2004). Once the dependency files were checked, any

source files which needed to be updated were compiled into object files, and the object files were immediately stored into the library file and removed. Once any libraries in need of update were compiled, the applications were examined. Each application that depended on a library that had been changed was in turn compiled. This overall structure ensured that only the files needed were compiled when a given change was made.

Within each function, a fixed structure was followed, to ensure that the pattern-based makefiles would be able to properly build the library. The root directory must have the name of the library, `vpd`, and must contain three subdirectories, one for header files, one for source code, and one for libraries. The directory structure for the VPD (Vertical Profile Display) is included in Figure 16 as an example:

```
funcs/vpd/
|-- inc
|   |-- _vpd.h
|   `-- vpd.h
|-- src
|   |-- draw.cpp
|   |-- draw_route_profile.cpp
|   |-- draw_symbology.cpp
|   |-- draw_terrain_profile.cpp
|   |-- funcs_vpd.dsp
|   |-- makefile
|   |-- on_click.cpp
|   |-- vpd.cpp
|   `-- wrapper.cpp
`-- lib
    |-- vpd.linuxd.a
    |-- vpd.linuxr.a
    |-- vpd.win32d.a
    `-- vpd.win32r.a
```

Figure 16 File Structure of the VPD library

There must be two header files in the include directory. One is named the same as the directory, `vpd.h`, contains the public interface of the VPD, and defines a C++ class named 'VPD'. The other, starting with an underscore, `_vpd.h`, contains the actual implementation of the function, and defines a class called "VPD_impl". This is the true implementation of the class, but is not visible to other functions in the project, which only see the public interface defined in `vpd.h`. This allows a simulation of Java's "class *x* implements *y*" metaphor of isolation of interface and implementation. This allows

developers to make changes to the implementation without those changes forcing a recompilation of all the various other functions that reference the VPD. As long as the interface to the VPD is not changed, there is no need to spend time recompiling large parts of the rest of the development tree simply because the internals of a commonly referenced library have changed. In the above example, the VPD class has only two member functions – a constructor, and a draw function. By contrast, the VPD_impl class implements the draw function as a series of calls, to draw_route_profile, to draw_terrain_profile, and so on. These steps in the overall draw process are split up into separate files for isolation from each other, but this split is hidden within the implementation, not visible in the interface, which exposes only the simple draw call.

The source code is split into one source file for each function call in the VPD_impl class. The constructor for the VPD_impl class appears in a source file named `vpd.cpp`. Finally, one additional source file, `wrapper.cpp`, serves as the definition of the public interface of the VPD. Its constructor creates a VPD_impl and stores it. Any subsequent calls to functions on the VPD class are simply passed along in turn to the VPD_impl. When the VPD class is destroyed, it first destroys its stored VPD_impl class.

When the library is built, it creates a number of library files in the lib directory. These files are named with the name of the class, combined with a tag indicating how they were compiled, and for what system. For example, a library built for Linux debugging of the VPD is vpd.linuxd.a, while a Windows release version is vpd.win32r.a

Data Storage And Arenas

Various data types are required to be loaded in memory in an area around the aircraft in order for the display to be useful. These include terrain data, navigational information, and weather data. Each of these is loaded into a number of objects of various classes that are derived from a generic point class. This generic point class contains functions to position an object on the surface of the earth. The source data is

stored in a variety of formats, some as a simple image file, and others as a complex relational database. Each of these data sources, however, is converted by a background thread task into useful data, and is placed by that server into a managed arena.

This managed arena examines the various objects in the arena, and ensures that old data is cleaned up when it is no longer useful to the display. Both loading and unloading of the data is managed without any direct intervention from the main thread, which must simply ensure the aircraft position is periodically updated. Then, the main thread simply has access to this available data, allowing it to render whatever data is there without regard to its management. Details of the management of the data are explained later, in the sections discussing the background server processes themselves.

Display Components

The main display of the application consists of four major components, the Primary Flight Display, Multifunction Display, Vertical Profile Display, and the Menus. These components are described in following sections.

Primary Flight Display

The Primary Flight Display (PFD), seen in Figure 17 on page 32, displays the aircraft attitude and primary flight information to the pilot. In this view, a synthetic view of terrain is seen, appearing as a sunlit brown surface, overlaid with a checkerboard texture. A path is also provided, which can be seen leading off to the left side of the display, then turning right just in front of the aircraft. Finally, aircraft parameters are visible, such as airspeed (150 knots), and altitude (13,535 feet). Several command boxes can also be seen, with commanded settings shown in magenta.

If the aircraft is equipped with an airmass based speed measurement system (pitot probe and air data computer), the display will show an indicated airspeed tape. Some low cost display systems show GPS derived groundspeed on a speed tape instead of airmass speed. It was felt that this is not desirable, because winds may cause significant

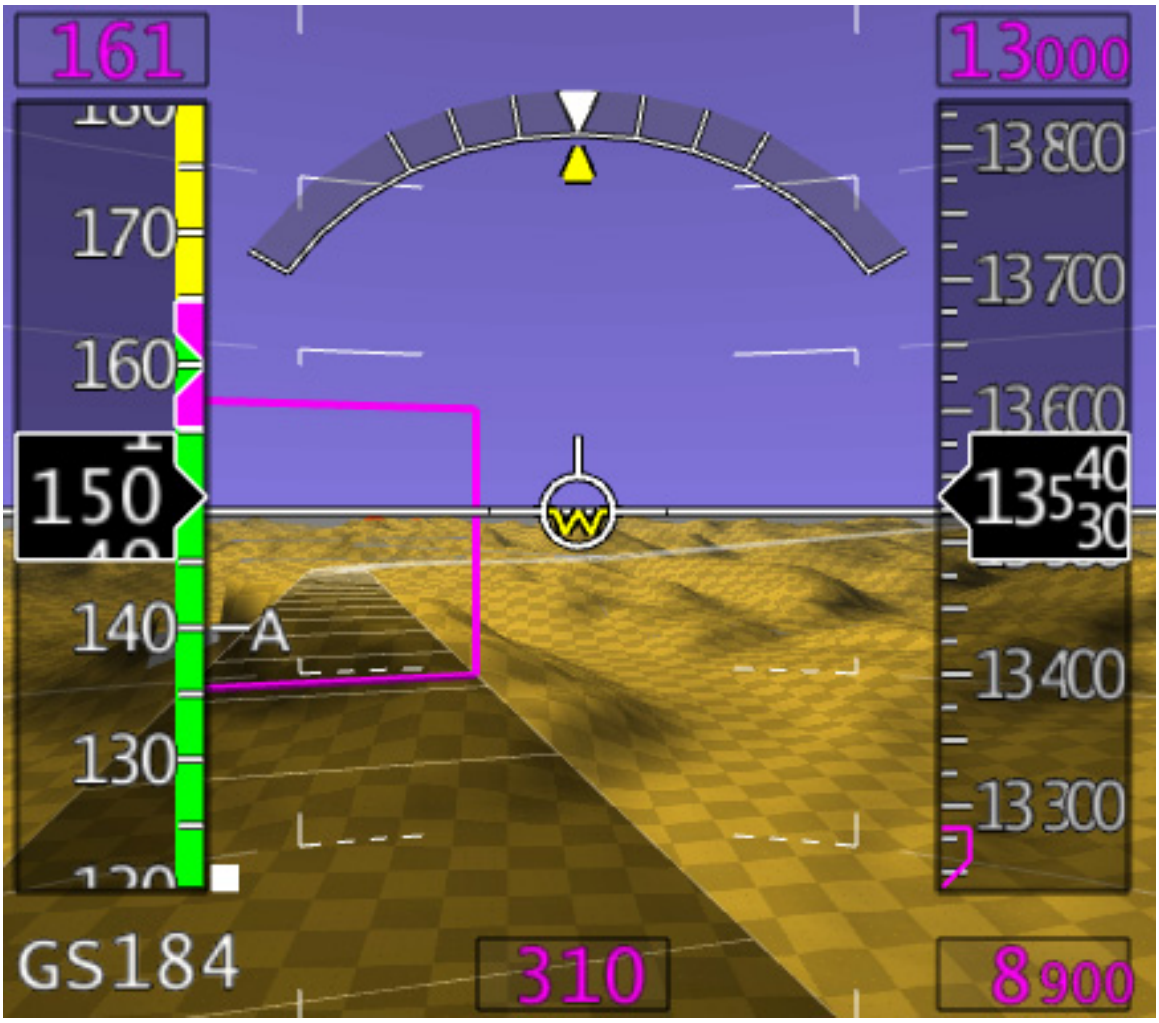


Figure 17 The SFB Primary Flight Display

differences between airspeed and groundspeed, which could mislead the pilot, and cause a stall, for example when flying with a strong tailwind. For this reason, we choose not to display groundspeed as a tape, but only as a digital readout. The PFD also includes an airmass based altimeter tape. Again, we feel that airmass data is better than GPS derived altitude data because the altitude tape is thereby referenced against the certified altimeter.

The PFD also contains boxes which display current commanded airspeed, altitude, and heading, and also allow them to be set, by touching the indications and entering a new setting on the keypad.

Finally, when GPS position is available, the PFD shows a depiction of terrain in front of the aircraft, along with any tower obstacles which are charted in the database. Terrain is shown as a brown surface with checkerboard overlaid. Obstacles are shown as gold pyramids with very steep sides. Terrain warning appears as a red overlay, highlighting dangerous terrain or obstacles for the pilot's attention. Airports and runways are also depicted, along with extended centerlines of each runway, allowing for easier alignment on approach. Finally, the active route is also drawn as a partial highway in the sky depiction on the PFD.

The software structure of the PFD module performs the steps illustrated below:

- Bind Graphics Context
- Position eye point
- Draw artificial horizon or standby indicator
- Draw Perspective Scene
  - Setup lighting, texturing, and culling
  - Draw Terrain
  - Draw Obstacles
  - Draw Warning Colors
  - Draw Airports
  - Draw Route
- Draw 3d Symbology (Pitch Ball and Flight Path Vector)
- Draw 2d Symbology
  - Draw Speed Tape and Altitude Tape
  - Draw Buttons

Bind Graphics Context – Each component of the display needs to communicate with the OpenGL subsystem to render its particular portion of the display. As the OpenGL engine is a state machine, there are various parameters of the rendering system which are set and maintained in a state, for example, whether texturing is turned on, or the active drawing color, or the position of the eye in the overall coordinate system. As it is not desirable to reset all of these parameters as each component of the display is drawn, instead a graphics context is created for each component. This allows maintenance of the state on a component-by-component basis. This allows for easier development, as it isolates change and effect within the component. Otherwise, for example, changing

texturing in the PFD might unintentionally affect the menus, and so on. In order to prevent this, the first step in drawing the PFD is to select the PFD's private graphics context.

Position Eye Point – All visual elements of the PFD are drawn in a single, consistent coordinate system. We have chosen ECEF (Earth Centered Earth Fixed) coordinates for all elements of the display. ECEF is a coordinate system with the origin at the gravitational center of the earth. The X axis in our chosen version of ECEF is projected out at the intersection of the prime meridian and the equator. This point is in the ocean, near Africa. The Y axis is projected out the intersection of the equator and the 90 degree east line of longitude. This point is in the ocean, near Indonesia and India. Finally, the Z axis is projected out through the North Pole. All units are measured in meters, along this right-handed coordinate system. To provide a sense of scale, the radius of the earth is about 6,371,000 meters, so a position vector in ECEF coordinates must have a length of approximately that number. It is also worth noting that this definition of ECEF is not universal, as some authors define the axes differently, usually by swapping the definition of X and Z, relative to what is described above. An illustration of the ECEF coordinate system is shown in Figure 18, on page 35.

Positioning the eye point involves taking the estimated position, using GPS and barometric data, and converting from traditional latitude, longitude and altitude to ECEF coordinates, and then passing these coordinates to the OpenGL subsystem. It also involves taking attitude data, expressed as pitch, roll, and heading of the aircraft, transforming that into appropriate orientation in the ECEF reference frame, and passing this to the OpenGL subsystem as well.

Draw Artificial horizon or standby indicator – A traditional blue-over-brown attitude presentation is visible as a bottom layer in the PFD, regardless of what terrain data is available or loaded. To this end, the first step is to draw this simple attitude presentation. A simple brown cone is drawn, at a very far distance from the eye, so that

any terrain which is loaded will later be drawn in front of this cone, covering it with a more accurate terrain model if it is available.

Setup Lighting, Culling, and Texturing – In preparation for drawing the terrain surface itself, a variety of accessory operations must be done. First, lighting must be positioned to effectively illuminate the terrain surface, allowing it to be effectively interpreted by the pilot.

Secondly, culling must be set up. As aircraft can change heading rapidly, terrain must be loaded in all directions around the aircraft, not just in its direction of flight. However, drawing all this terrain, even behind the aircraft, would be a tremendous waste of resources. One way of dealing with this problem is to use culling. The viewable area

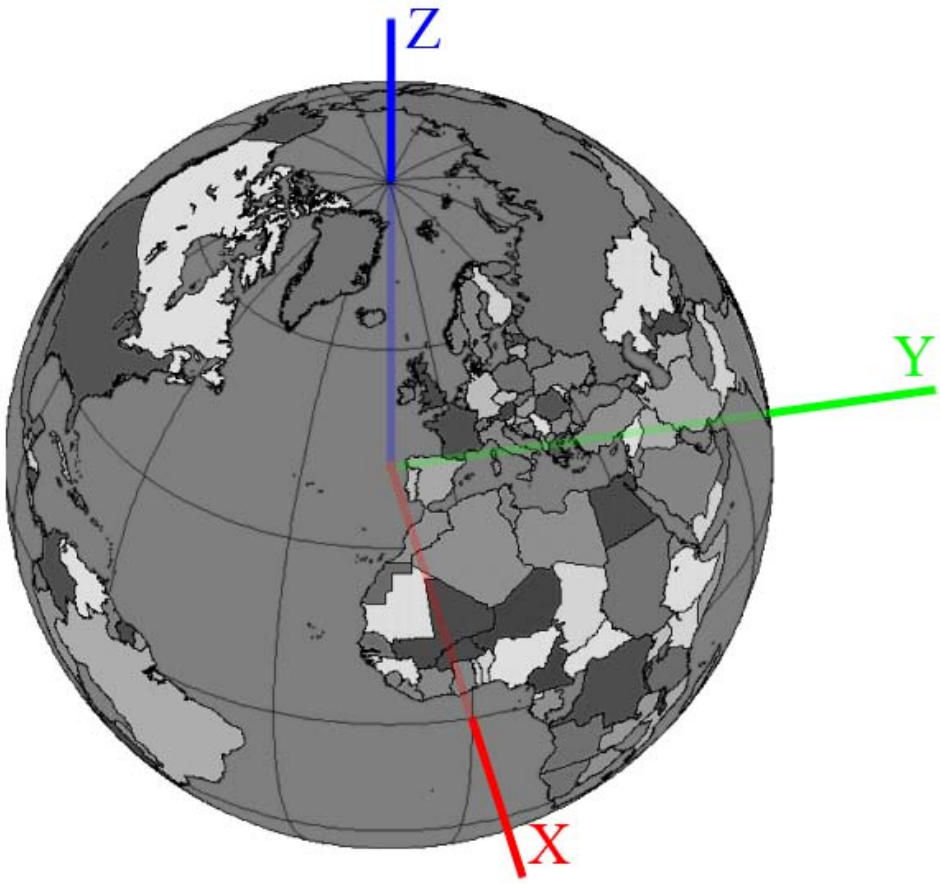


Figure 18 ECEF Coordinate Axes

ahead of the aircraft is computed, as a collection of six equations of a plane which specify the edges of the top, bottom, left, and right edges of the screen, as planes oriented in ECEF coordinates. Also, a near and far bound on distance from the eye is established. Each of these planes is considered to have an 'inside' and an 'outside'.

Objects are considered as having a center point, and a sphere which completely bounds the object. Any object whose bounding sphere lies completely 'outside' any one of those six planes will not appear on the display, so it is safe to skip over drawing it. Objects whose bounding spheres straddle, or are inside of all six planes may appear on the screen, therefore must be drawn. An illustration of this can be seen in Figure 19. It is possible to cull approximately three-quarters of the terrain around the aircraft, reducing rendering costs.

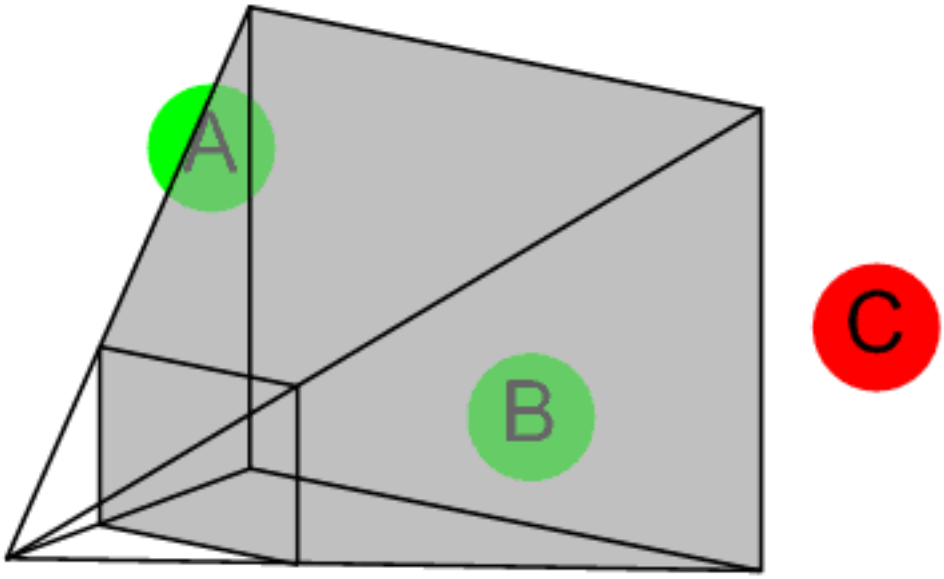


Figure 19 Frustum Culling

Note: The area visible on the screen is a truncated pyramid. The eye is considered to be at the pointed tip of the pyramid, and the volume visible on screen, called the frustum, is shaded in grey. Objects very close to the eye are not visible, nor are areas far from it. Sphere A is partially inside the frustum, so portions of the sphere will be shown on the screen. It needs to be drawn. Sphere B is completely inside the frustum and also needs to be drawn. Sphere C is completely outside the frustum, and can be skipped to save rendering costs.

Finally, texturing is configured. The terrain is textured with a simple checkerboard pattern. This pattern is highly effective in allowing the pilot to judge proximity to terrain, as the size of the checkers grows larger as they approach the terrain surface. The regular grid pattern also assists the pilot in knowing their orientation relative to cardinal directions. It is set as a two-dimensional texture, aligned to north and east, with a center point that is moved to remain close to the aircraft.

Draw Terrain – The terrain itself is rendered as a collection of tiles. Each of these tiles has a center point which is positioned and oriented relative to the eye, and the remainder of the tile is drawn relative to the center point. Each tile's center point and bounding radius is checked against each of the six culling planes. If it is inside all six planes, then it is drawn, otherwise it is skipped. Each tile's distance from the eye is computed, and is drawn with progressively less detail, with increasing distance from the eye. Tiles consist of a surface mesh of points which are rendered as triangle strips. Each point also is associated with a surface normal, which is used to create accurate lighting.

Draw Obstacles – Obstacles are point features with an associated elevation. Each obstacle that is loaded is tested against the same culling planes as the terrain has been tested. Each obstacle that passes is rendered.

Draw Warning Colors – Warning colors are used to highlight terrain and obstacles that are dangerously close to the aircraft's projected flight path. The flight path chosen is a two-step path. The aircraft's path is projected forward sixty seconds into the future. At that point, an immediate climb of five hundred foot per minute is projected. Any terrain or obstacle that is within 500 feet, vertically, of that projected flight path, is highlighted with a red tint, as a warning to the pilot. The reason for this segmented path is that the traditional method of resolving a terrain or obstacle conflict is to climb to higher altitudes. Given the poor historical accuracy of ground surveying and aircraft position estimation, terrain warning systems do not support maneuvering or turning to avoid obstacles. The path, seen in Figure 20, on page 38, represents the future actions of

a pilot who, while descending into hazardous terrain, is alerted to the terrain conflict, and climbs to escape the danger. The sixty seconds of projected descent allows adequate time as a margin for the pilot to initiate a maneuver to escape the potential terrain conflict.

Draw Airports – Airport data is loaded from the database. Each loaded airport is drawn in this step, as a collection of runways and runway markings. These features are all referenced to a defined center point of the airport, a surveyed point called the Airport Reference Point. As with terrain and obstacles, this airport reference point is compared to the clipping plane. Unlike terrain and obstacles, however, the bounding sphere of each airport is of a different size, commensurate with the overall size of the airport. If the airport is found to be visible, the runways are rendered in successive passes. First, the white outlines of the airport's runways are rendered. Second, the black runway surfaces are rendered. Finally, the runway centerlines and markings are rendered. Each runway end features standard approach markings, along with runway numbers and extended centerlines.

Draw Route – The depiction of the route is relatively expensive. Because of this, most of the work involved in rendering it is computed and stored, each time the route changes. The stored geometry for the route is rendered, again with reference to the standard ECEF coordinate system. Details of the route pre-computation will be covered

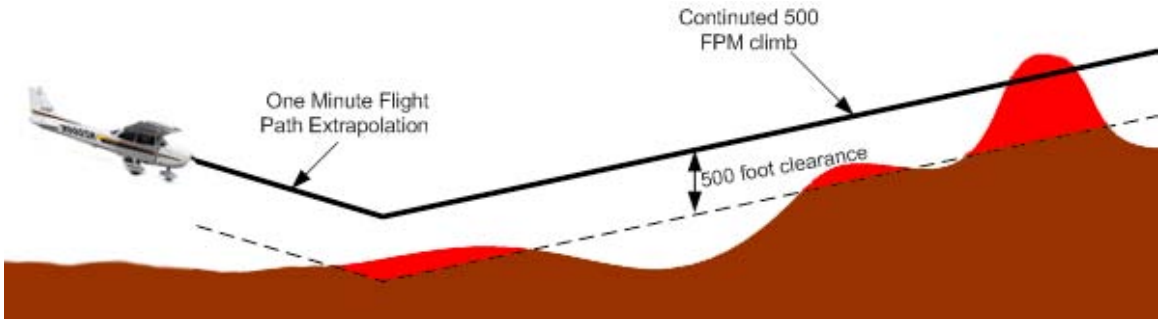


Figure 20 Cross section showing segmented warning path

Note: Red areas in this figure are highlighted on the PFD when viewed in perspective.

in the section specific to the route.

Draw 3d Symbology – Some portions of the display are best rendered as fully three-dimensional objects. As it is very important for the pitch ladder to match and be conformal to terrain and obstacles to allow estimation of angles, it is rendered as a ball around the pilot's eye point. This may be seen in Figure 21. The flight path vector, indicating the direction of travel of the aircraft, is similarly rendered.

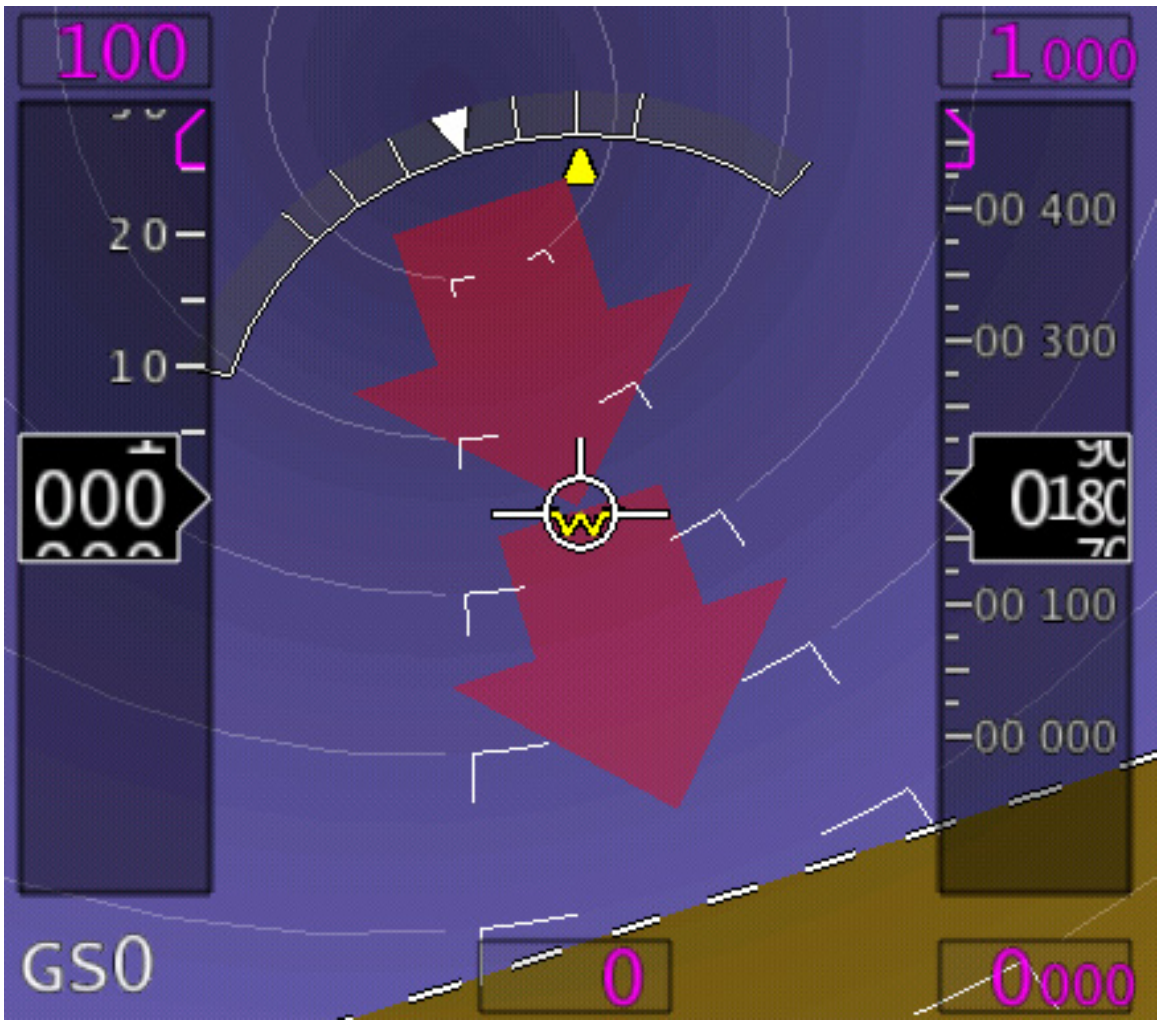


Figure 21 The PFD during an unusual attitude situation

Note: The curved lines of pitch become full circles near the zenith. Note also that the horizon line has become dashed, and a false ground fill has been added, to cue toward the direction of terrain that is not visible due to the extreme pitch of the aircraft.

Draw Speed Tape and Altitude Tape –The speed and altitude tapes, due to their common behavior of a scrolling tape behind a fixed window showing current values, are programmed as variations on a common framework. Each tape is customized in terms of the number of digits displayed, the range of the visible tape, and so on. Each tape is drawn in a simple two-dimensional coordinate system, measured in millimeters on the display surface, allowing for easier configuration and adjustment to a specific display setup.

Draw Buttons – Several buttons appear on the PFD, and serve roles both as indicators of commanded flight parameters and as a method to set those commands. For example, above the altitude tape, a button is shown indicating the currently set commanded altitude. This commanded altitude also sets the command bug on the altitude tape, which helps the pilot maintain that altitude. If the user touches that button, they are prompted to enter a new command. These buttons are a specific instance of a more generic button class used throughout the application as a whole. Like the speed and altitude tapes, they are drawn in a simple 2D coordinate system.

Multi Function Display

The Multi Function Display (MFD), seen in Figure 22 on page 41, serves as a moving map, centered on the aircraft position. It provides the pilot with strategic information such as nearby terrain, weather, navigational radio aids, and features of the airspace system. The aircraft's planned route can be seen in white, as can the angled grey bar of the Course Deviation Indicator (CDI), used for guidance on the route.

As with the PFD, a fixed set of operations are performed to draw the MFD. Unlike the PFD, the user has control over what display elements are drawn or not drawn. For example, the pilot may chose to clear the screen of certain display elements such as navigational aids, runway extension lines, etc., if they are cluttering up the MFD. Also, though many things may appear on both the PFD and the MFD, the exact method of

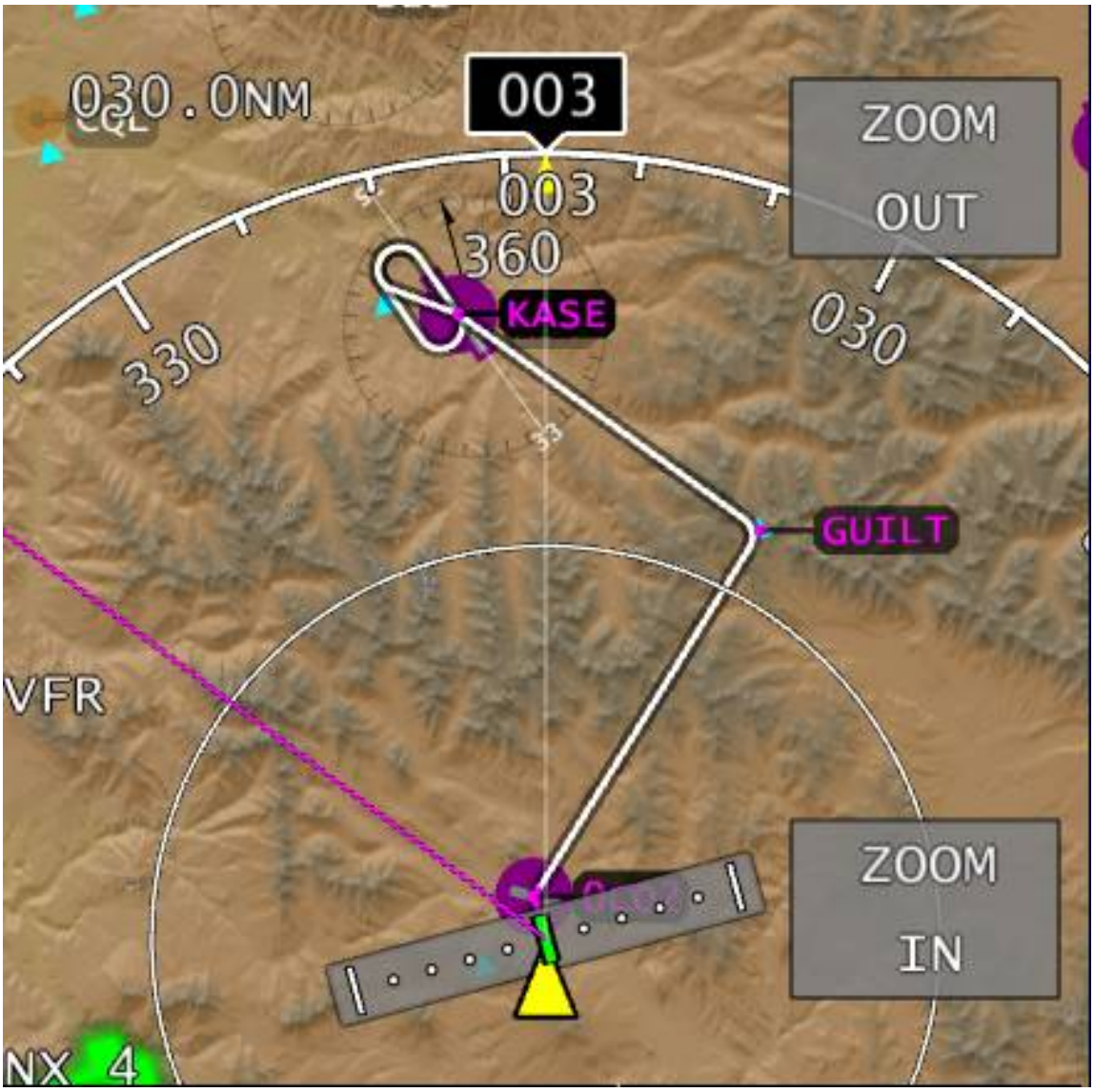


Figure 22 The SFB Multi Function Display

drawing these objects may be completely different, due to the necessities of a differing presentation. The overall set of operations which must be performed to draw the MFD are as follows.

- Bind graphics context
- Draw arena and route
  - Setup eye point and culling
  - Draw terrain and NEXRAD weather
  - Draw airspace boundaries

  - Draw airways
  - Draw VORs and NDBs
  - Draw Waypoints
  - Draw Airports
  - Draw proposed and active routes
- Draw ownship, CDI, Compass Rose, and Annunciators
- Draw Keypad

Bind Graphics Context – As with the PFD, each component of the display maintains its own copy of the OpenGL state, allowing it to be kept separate from other components. The first thing the ND does is to bind its own private graphics context.

Draw Arena and Route – As with the PFD, the first step is to position the eye. The same ECEF coordinate system is used to render the moving map display. An orthographic projection is used. At large map ranges, this means that the map features depicted appear as they would when observing a round globe. This of course causes some distortion, as points very far from the aircraft become more and more compressed, but it presents a very natural and easy to interpret view of the map. Varying map ranges are drawn by adjusting the viewable area, sent to the OpenGL subsystem as a set of projection parameters. As with the PFD, culling planes are computed, to limit what is drawn, once all of the necessary display setup is finished.

Once the eye point is positioned, various arenas are serviced. In each case, the display configuration is first checked to see if the given element is set to be displayed or not. If it is, each point is checked against the culling planes. Each class of object has a defined center point and bounding sphere radius, and is only rendered if some portion of it on the screen. Each is rendered as a shape referenced to its center point.

Terrain and NEXRAD tiles are both rendered quite similarly. Each tile is drawn as a simple rectangle, representing an area of the earth's surface, with an appropriate texture mapped onto it. It is not necessary to render the full geometry of the terrain, as vertical features are not visible in the map display. Terrain tiles are rendered in one of three color schemes, though all three schemes are combined with a bump-mapped

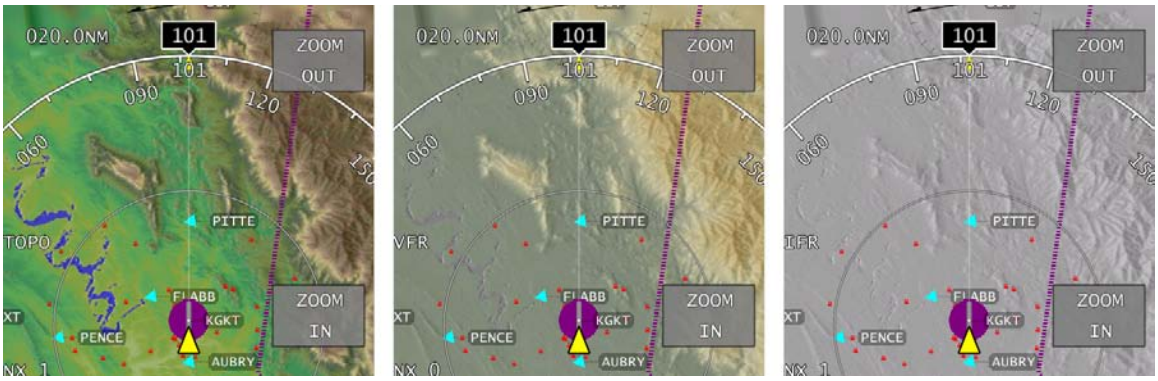


Figure 23 The terrain color schemes

Note: Left to right are the Topo, VFR, and IFR color schemes.

texture, allowing for easy interpretation of the underlying terrain shape. The first color scheme, for IFR use, is a uniform grey color, allowing for minimum interference with the presentation of radio navigation aids and other airspace features. The second scheme is a VFR scheme which uses a similar color scheme to standard VFR sectional charts, already familiar to pilots. The third theme is a topological color scheme, which highlights terrain features even more strongly, but may somewhat obscure the map's radio navigation and airspace information. All three of these color schemes may be seen in Figure 23.

NEXRAD tiles are drawn in a similar manner to the terrain tiles, but are laid overtop the terrain as a partially transparent layer, using standard colors for airborne weather. Each tile is referenced to the ground by its center point and extents, and is coded with color to indicate the intensity of the weather returns seen in the data-linked information. Light and dark green represent light precipitation, light and dark yellow represent moderate precipitation, light and dark red represent heavy precipitation and magenta represents extreme precipitation. For the purposes of a general aviation aircraft, yellow returns include rain and turbulence sufficient to ensure an uncomfortable ride, and any red or stronger returns represent areas which pose a real threat to the safety of the aircraft. An example weather display may be seen in Figure 24 on page 44.

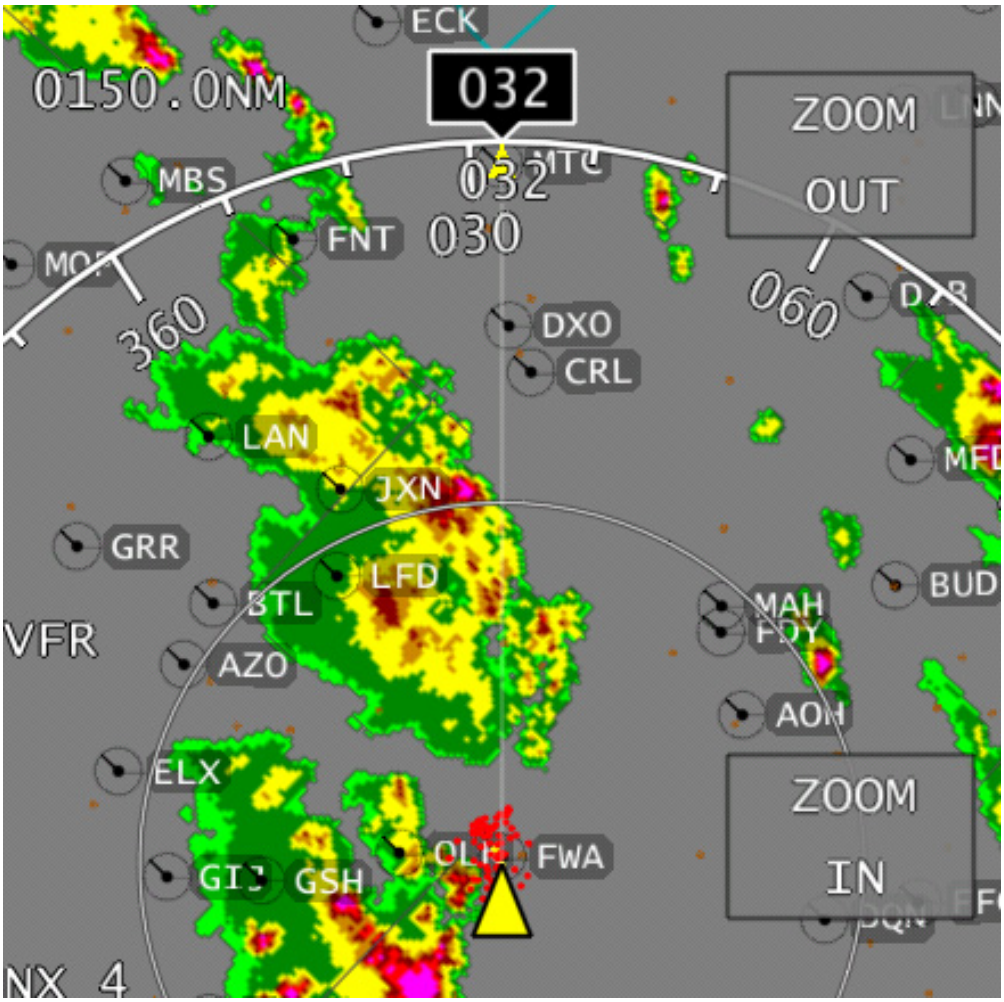


Figure 24 NEXRAD weather on the Multi Function Display

Airways are drawn as simple straight-line segments, connecting various radio navigation aids and waypoints, as they are stored in the database. Each is referenced to a center point at the midpoint of the airway.

Airspace boundaries are loaded as segments, either straight-line or constant radius arcs. While simple individually, a collection of these straight or curved segments can depict a much more complicated airspace boundary. Each is again stored as either a straight-line segment, referenced to its center point, or as a collection of straight line segments, approximating the curved arc, referenced to the center point of the arc.

VORs, NDBs, and waypoints are each drawn as simple point features, with an appropriate symbol showing their position. VORs and NDBs, in accordance with their traditional paper chart depictions, indicate the direction of magnetic north. Waypoints are a simple triangle, as they are on IFR charts.

Airports are drawn using their VFR chart equivalent presentations, illustrating their center point, along with a symbolic depiction of their runway arrangement. As with 3D drawing on the PFD, runways are drawn relative to the Airport Reference Point.

Each of the navigational information types mentioned above is automatically removed from the display as the map range is increased, to prevent the display from being flooded when large map areas are viewed. The ranges at which this happens for various display elements may be seen in Table 2. Also, each feature being drawn on the display may draw a text label identifying it, depending on the map range, chosen to avoid excessive clutter. No attempt is made to separate labels if they overlap due to close proximity or collocation of various features. In the event this occurs, the labels are drawn with a priority order, with labels of more important objects covering the labels of less important objects.

Finally, proposed and active routes are drawn. Much like the PFD, each is drawn primarily from a script that is computed as the route is changed, rather than computed for each frame. Waypoints along the route are labeled by name, regardless of the map range chosen, and route waypoint labels will always cover any other label types.

Table 2 Ranges at which elements of the Multi Function Display are hidden

Display Element	**Maximum Range Drawn**
Terrain	Always
Weather	Always
Airways	250 nm
VHF and NDB navigation aids	250 nm
Waypoints	100 nm
Airports	60 nm

Draw ownship, CDI, Compass rose, and annunciators – As with the PFD, each of these features are drawn in a simpler 2D coordinate system, with sizes specified in millimeters, for easy configuration. The ownship symbol is a simple triangle shape, and serves as the center point of the CDI (Course Deviation Indicator), when it is drawn. The CDI is drawn whenever the aircraft has an active route and is considered to be on course. In this situation, a translucent box is drawn, perpendicular to the desired track, with a line indicating side-to-side deviation from the centerline of the desired course. See Figure 25. The presentation is very similar to a standard HSI, with the exception that the display can also guide through curved segments such as DME arcs. The full width of the scale from edge to edge is 2 nautical miles. Each dot on the scale represents $^{1}/_{5}$ of a nautical mile.

Finally, annunciators are drawn to provide feedback to the pilot when elements of the display are not drawn. Each is depicted as a text label, crossed out when the matching element is disabled. When enabled, the annunciator clears.

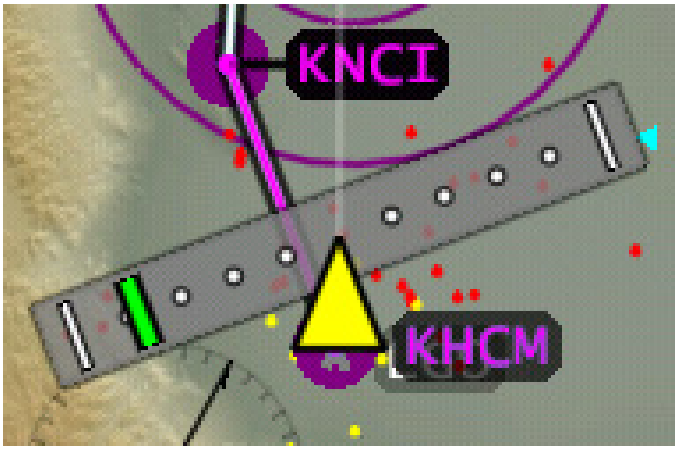


Figure 25 The SFB CDI, as part of the MFD.

Note: The desired course is shown in magenta, from KHCM to KNCI. The shaded bar can be seen to be perpendicular to the path. If the CDI is level on the screen, the aircraft is aligned to the path. The green mark on the CDI indicates where the path is relative to the aircraft – The centerline of the desired path is approximately .8 nm to the left of the aircraft, and the angle of the CDI indicates the aircraft is flying away from the path. To intercept the path, the pilot needs to turn left.

Draw Keypad – Several elements of the display need to obtain input. The PFD is used to configure commanded airspeed, altitude, and headings. Each of these is entered on a keypad. The normal place for this keypad was chosen as a telephone-style keypad overlay on the MFD. Also, identifiers of points to be added to the route are entered on the same keypad, spelled out as letters on the same telephone-style keypad. The telephone star button is replaced on this keypad with a backspace button, while the telephone pound button is replaced with an enter key. An area is left on the enter key for a preview of the entered text, to allow for the user to verify their inputs. The keypad overlay can be seen in Figure 26.

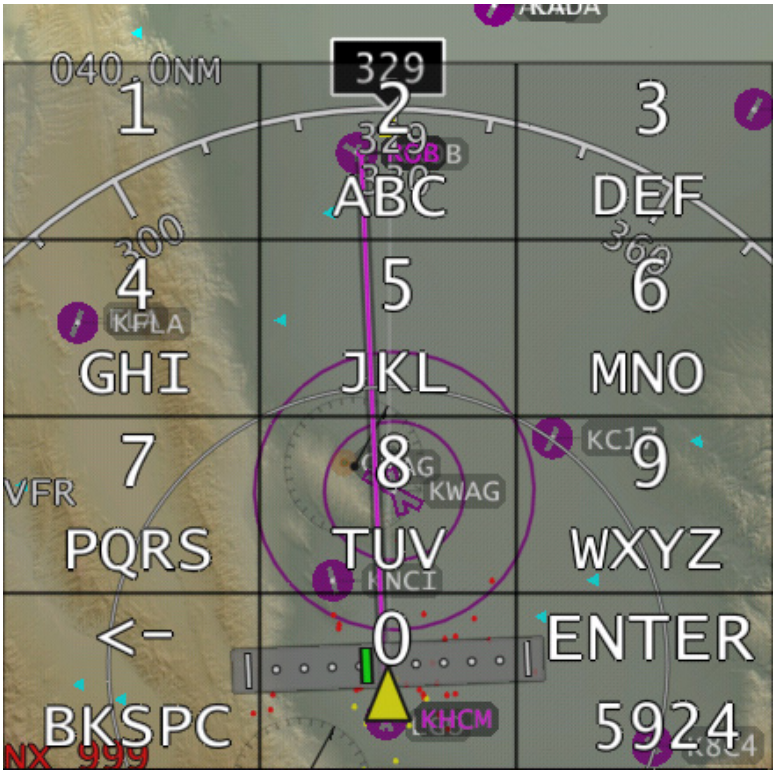


Figure 26 The keypad overlaid on the MFD

Note: The keypad is drawn semi-transparent over the MFD, when in use. In this case the pilot is adding a waypoint at KWAG. They have pressed the 5JKL key once, the 9WXYZ key once, the 2ABC key once, and the 4GHI key once. The display shows 5924 on the enter key, to allow verification.

Vertical Profile Display

The Vertical Profile Display (VPD), seen in Figure 27 and Figure 28, is a cross section of the earth's surface in front of the aircraft, sliced to show the profile of its surface. It is rendered to allow the pilot another preview of the approaching terrain. Also drawn is a profile of the upcoming flight plan, allowing effective planning of climbs and descents, and estimation of distance to upcoming waypoints. It is drawn in a number of steps, illustrated below:

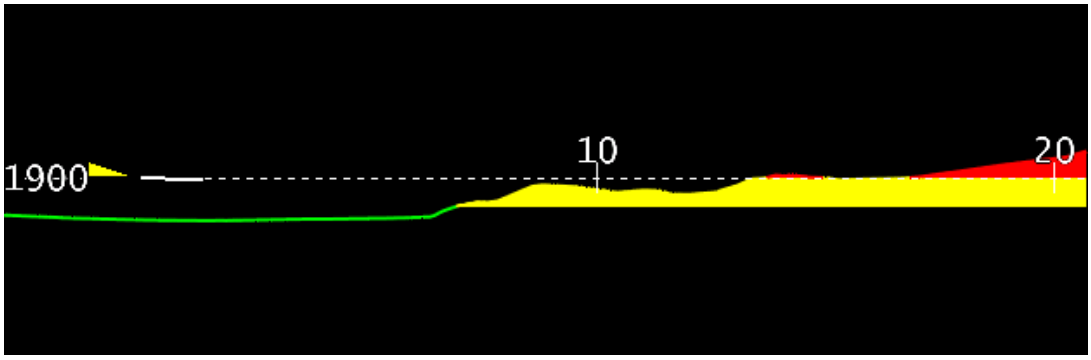


Figure 27 The Vertical Profile Display, while approaching rising terrain.

Note: Terrain shaded yellow or red is designed to alert the pilot to a potential hazard.

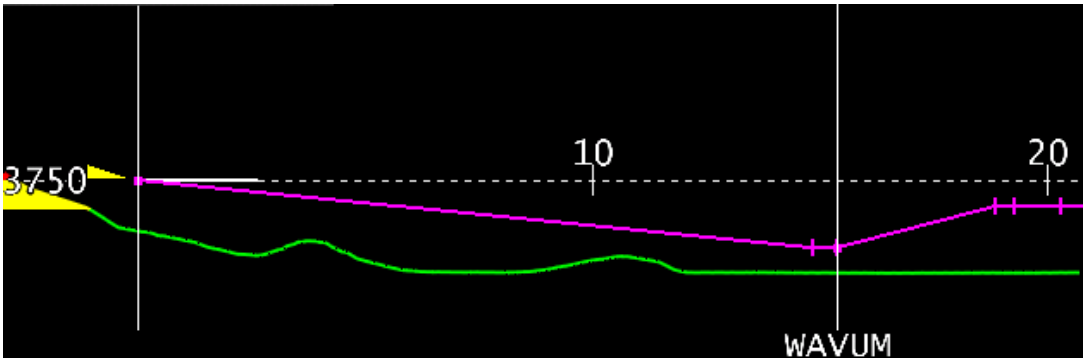


Figure 28 The Vertical Profile Display, with a route depicted.

Note: The magenta route indicates a descent, which finishes just before the waypoint WAVUM is reached. Starting at WAVUM, a climb occurs. WAVUM can be seen to be approximately 15 miles away. The end of the climb is approximately 18 miles away.

- Bind Graphics Context
- Compute Terrain Profile
- Draw Terrain Profile
- Draw Symbology
- Draw Route Profile

Bind Graphics Context – As with the PFD and ND, the local state of the OpenGL subsystem is preserved and isolated in a graphics context. It is bound at the start of processing.

Compute Terrain Profile – The VPD represents a cross section area ahead of the aircraft, and as such has a certain width associated it. It is divided into a number of evenly spaced slices, perpendicular to the flight path, with increasing distance from the aircraft. Along each of these slices, there will be a variation in height, and it is desirable to know both the lowest and highest altitude in that area of terrain. In order to do this efficiently, it must be possible to sample the terrain at various points. This sampling is nontrivial, as it requires a grid of points to be found, with axes in the direction of flight and perpendicular to it, while terrain data is stored as a grid aligned with lines of latitude and longitude. To do this requires transforming the terrain into a local projection relative to the aircraft and sampling the projected grid. Some portions of this projection may be accelerated using features of the OpenGL subsystem.

Draw Symbology – The VPD has a limited amount of fixed symbology, primarily the ownship symbol, a yellow triangle akin to the ND, and a white line indicated future position. The start of the line is at the current aircraft position, and its end indicates the extrapolated position of the aircraft, predicted sixty seconds into the future. This allows the pilot to predict both speed along the ground, and rate of climb or descent. As combined, the angle of climb or descent is also presented, allowing for the pilot to judge if a climb can be completed in time to clear a future obstacle, or if a descent can be completed in time to reach approach altitude before reaching the airport.

Draw Terrain Profile – Data that has been computed and stored in the previous steps is now rendered. The presentation used is that terrain more than a thousand feet below the aircraft is shown in green, while terrain less than a thousand feet below the aircraft is yellow, and terrain above the aircraft is red. Above the terrain profile, any empty area of the screen is colored black, to allow for better contrast with other portions of the symbology.

Draw Route Profile – The route is rendered as the last portion of the display. When a flight plan has been entered and the aircraft is considered to be on route, a side-view of the route ahead of the aircraft is presented. This side view depicts the route as it would appear if straightened, as it is not possible to accurately depict the route with curves when viewed from the side.

Menus

The menu system was designed from the beginning to be flexible, yet fit with a number of constraints. Button size was dictated by the amount of space needed both to write clearly legible labels, and to easily select with a finger. Selection of items from the database was to be done using the same space occupied by the menus, as is a display of the route needed in the same space. For this reason, the ability to show an ordered sequence of buttons was another requirement. Finally, an important user requirement was found in that many users of the touch screen prefer to rest or brace their fingers along the right side of the display and press the buttons with a thumb, especially in turbulence. This leads to a preference for placement along the right edge of the display. For this reason, a layout of a single column of buttons along the right side of the screen was chosen. Based on the size of the screen, a 12-button menu was chosen. The structure of the menus was laid out for this number of buttons, and grouped appropriately.

Within the menu, a number of individual buttons are created in a large pool that is not grouped by function within the menus. There is also a 12-element list of buttons to

be displayed at any given moment. Each of these buttons can be associated with a button from the pool, based on the menu's state – In essence, a record of which menu the user is currently in. Button presses in a given state will either perform actions in the software, or change the state of the menus by moving to a different menu, or both.

Each state has a number of functions associated with it. The first is a setup function. Given any starting state, the setup function will clear the buttons currently displayed, and then reassign the buttons to the set of buttons assigned to that particular state. The second function is a handler for button presses. This press is first intercepted by a function of the menu that listens for the press, and receives the x and y coordinates of the press on the screen. These coordinates are mapped into buttons and recorded both as both a number, indicating which position in the menu the button occurs in, and also as a pointer to the button that was found in that position. Based on the state of the menu, the appropriate handler function is then called, with those two bits of information passed in. At that point, the function determines the set of operations that is to be performed on any given button press.

As an example of this, consider the root menu. The setup function for the root menu is called whenever the user returns to the root menu from some other menu. It is also called when the menu is displayed for the first time. The setup function clears whatever buttons are placed, and replaces them with buttons from the pool. Buttons added are labeled BRF/PRFLT FLP, DISP, Direct To, Gradient up, Gradient down, and so on. When a button press is detected, the menu finds which button on the menu was the one pressed. At this point, this button is passed to the handler function for the main menu. The handler function compares the pressed button to each button it knows the menu contains. If the button is the BRF/PRFLT button, then the menu is changed to the BRF/PRFLT menu, if the button is the FLP button, then the menu is changed to the FLP menu, and so on. In any of these cases, the menu change will call the setup function of the appropriate menu. By contrast, if the button pressed is the Gradient up button, then a

set sequence of actions will be taken, whatever is necessary to perform the actual command. In this case, the menu will stay as the root menu, rather than changing. A similar set of actions happens regardless of the menu as a whole.

A special mode is used for displaying lists of objects. In this case, buttons are created which are associated with the individual objects. A special setup function is called when it is desired to display a list of objects. This function manages the repeated work of positioning an appropriate set of buttons into the menu in sorted order, and supports scrolling, If more buttons are available than will fit on the list. This function is used both to display the route for interaction, as well as to select navigational aids which have been requested from the database.

Illustrations of the menus and their usage are included in the Functions section, starting on page 68.

Route

Perhaps surprisingly, management of the route is one of the most complicated aspects of the overall project. This complexity is managed through the implementation of layers, each representing an increasingly detailed description of the overall route.

Route Structure

At the highest level, the route is described as a starting point and an ending point. No route may be created without both. It also consists of five sections, each of which consists of a number of legs. The five sections of the route are a departure procedure, an enroute section, an arrival section, a transition section, and an approach. Of these, the only the enroute section is directly editable by the user, allowing a flight plan to be entered. The legs in the other four sections may only be selected as a sequence of legs which are grouped together in a published database of terminal procedures.

Within each section of the route, the desired path is represented by a series of legs. A leg is defined most simply by a fixed starting point and a fixed ending point, and

connects the points with a straight line. This is the type of leg which can be created by the user, as part of a custom-entered route. No discontinuities are allowed in the list of legs. In other words, the ending point of each leg must be identical to the starting point of the following leg.

Legs which are part of the terminal procedures database may represent more complicated paths, and may only be created via being loaded from the database. For example, legs may be created that connect the points not with a straight line, but with a constant-radius curve. In this case, an additional point is needed to define the center point of the arc. Alternatively, terminal legs may have an end point that is defined not as a fixed position, but rather as a function of their start point and some other parameters. With the exception of the holding pattern leg, these complex legs may not be directly created by the user, but rather only loaded from a database. The possible leg types are enumerated below, and can be seen in Figure 29 on page 55.

Standard Leg: The leg starts at a fix which is known to be at a defined position and ends at another fix, again with a defined position. If no center point is provided, the course is a straight line connecting those two fixes. If a center point is provided, then the course is a curve which is a segment of a circle centered on the center point, starting at the start fix, and ending at the end fix. If a center point is provided, it must be equidistant from the start and end fixes. This type of leg may be created by the user in the enroute section.

Course to an Altitude: The leg starts at an undefined point, wherever the last leg ended. The aircraft travels with a specified course along the ground, climbing until a specified altitude is reached. When that altitude is reached, the leg ends at that point.

Course to a Fix: The leg starts at an undefined point, wherever the last leg ended. The aircraft turns to the specified course, and flies along that course to the specified fix. The leg ends at that fix. The starting point is expected to be on the given

course line from the fix, but small corrections may be necessary to ensure that alignment. These corrections are done at the start of the leg.

Course to an Intercept: The leg starts at an undefined point, wherever the last leg ended. The aircraft turns to a specified course, and flies along that course until it intercepts a line, which is defined by a specified fix, and a specified bearing line from that fix. When the line is intercepted, the leg ends at that point.

Fix to a Course: The leg starts at a specified fix, which is known to be at a defined position. The aircraft turns to a specified course, and flies along that course for a specified distance, or for a specified time. When that distance has been traveled, the leg ends at that point.

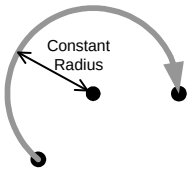
Procedure Turn: The leg starts at a specified fix, which is known to be at a defined position. The aircraft turns to a specified course, and flies along that course for a specified distance. When that distance has been reached, the aircraft turns forty five degrees, either to the left, or to the right. The turn direction is specified. After turning forty five degrees, the aircraft travels along a straight line for a time, and then turns one hundred eighty degrees in the opposite direction from the initial turn. At this point, the aircraft flies straight until it intercepts the course line that was originally flown, and then turns in to intercept the original course, now flying in the exact opposite direction. When the inbound course is intercepted, the leg ends at that point.

Holding Pattern: The leg starts at a specified fix, which is known to be at a defined position. The aircraft turns, and flies a holding pattern, referenced to the specified fix. The inbound leg of the holding pattern is on a specified course, and turns are made either to the left or the right. The turn direction is specified. The leg ends at the same fix at which it started. This type of leg may be created by the user in the enroute section.

Standard Leg (Straight)

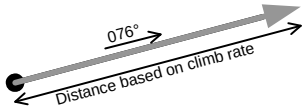
Starts at a defined point.
Ends at a defined point.

Standard Leg (Curved)



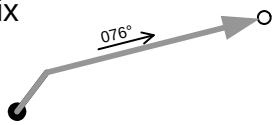
Starts at a defined point.
Ends at a defined point.
Forms an arc around a defined center point
Has a specified turn direction (clockwise/counterclockwise)

Course to an Altitude



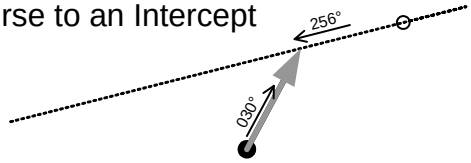
Starts at a defined point.
Travels along a defined course. (076°)
Ends after a computed distance, based on climb rate.

Course to a Fix



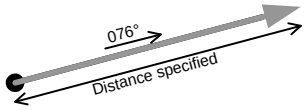
Starts at a defined point.
Corrects to bring defined course into alignment with fix
Follows defined course toward fix
Ends at defined fix.

Course to an Intercept



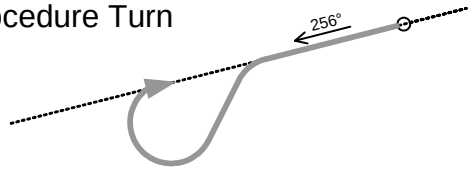
Starts at a defined point.
Travels along a defined course. (076°)
Ends when 256° radial is intercepted.

Fix to a Course



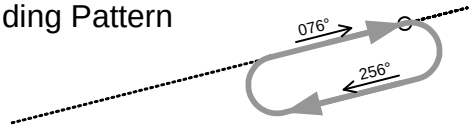
Starts at a defined fix.
Travels along a defined course (256°)
Ends after a specified distance.

Procedure Turn



Starts at a defined point.
Travels along a defined course (256°)
Executes a standard procedure turn
Ends on the inbound course (076°)

Holding Pattern



Starts at a defined fix
Turns 180 degrees.
Travels outbound (256°)
Turns 180 degrees.
Ends on the inbound course (076°)

Figure 29 Leg types used in SFB navigation

Note: Only the first type and the last type, (Standard straight leg and Holding Pattern) may be created by the user. The rest may only be defined by a navigation database.

These collections of legs allow the path along the ground to be defined, but make some approximations. For example, changes in direction at a waypoint are considered to be instantaneous, rather than a smooth curve as must be flown. To create this next level of detail, the various sections of the route are combined together in sequence and each leg of the combined route is converted into one or more segments. Segments, like legs, may consist either of straight line paths between two points, or constant-radius arcs. Once converted from legs to segments, the path will be smooth and continuous, having no abrupt changes of direction. Each segment that is created is considered to have come from one specific leg only. This is necessary as it must be possible to determine which leg of the flight plan is active, based on which segment of the flight plan is being flown.

Finally, once this collection of segments that provides a smooth representation of the path is computed, some individual segments are split into multiple segments, based on what changes in altitude may be necessary as part of the flight plan. For example, if a climb is begun at the start of a long segment, such that the climb may be finished before the end of the segment is reached, the point where the climb is finished is computed, and the segment is split into a climb portion and a level flight portion.

At the lowest level, the list of segments is turned into a special purpose data format that is more suited to direct display via the OpenGL subsystem. Each segment is converted into a collection of small straight-line path sections which adequately represent the climbs, descents, straight-line regions, and curves of the segments. These path sections are then stored, to be used for the purpose of rendering the route on the PFD.

Route Changes

Each change to the flight plan that is performed is performed at the highest level, that of the list of legs. Several operations are allowed, each of which modifies the lists of legs. When this modification is complete, the segments are recomputed from the legs, to bring them back to a consistent representation. The allowed operations are:

- A new route may be created, given a start and end point.
- A new point may be added to the enroute portion, given a point to add, and a point to add the new point before. If no point is provided, the new point is added at the very end of the route.
- A point may be removed from the enroute portion, given the point to be removed.
- A hold may be added to the enroute portion, given a point to hold upon that is already in the route, along with an inbound course to hold upon, and a direction for the turns to be made while holding.
- A hold may be deleted from the enroute portion, given the holding point which is already in the route.
- A point in the enroute portion may be replaced by another point, given a point to be replaced, which must already be in the route, and a point to replace the original point with.

Each of the above operations may be done as a way of editing the enroute portion of the route. Each of these is an operation on points in the route, and is turned into a set of operations on legs as needed. For example, if the route connects a series of points, A-B-C-D, and an operation is commanded to add a new point X between B and C, the actual effect within the route is the removal of the B-C leg, replaced by two legs, one B-X, and one X-C. In this way, the pilot can think in natural terms of waypoints, while the software represents the route in terms of legs connecting those waypoints.

Additionally, changes can be made to the other portions of the route, though only in a more constrained manner. As operations in the terminal environment are strictly defined in published procedures, it is important to be able to add these procedures to the route, and ensure that they not be accidentally changed or modified by the pilot once loaded. To that end, the following more restrictive operations are allowed:

- The departure procedure may be set to a defined list of legs. Setting the departure procedure to a blank list of legs clears it.
- The arrival may be set to a defined list of legs. Setting the arrival to a blank list of legs clears it.
- The transition may be set to a defined list of legs. Setting the transition to a blank list of legs clears it.
- The approach may be set to a defined list of legs. Setting the approach to a blank list of legs clears it.

Route Computation

Whenever the list of legs in the route is changed, the next step is to convert the list of legs into a list of segments. To do this, the list of legs is first scanned until the active leg is found. If the active leg is found, the start of the active leg is set to match the same elevation as the aircraft's current height. This ensures that the aircraft will be vertically centered in the route when it is modified. Otherwise, climbing or descending while changing the route would cause the aircraft's position in the climb to be reset and lost.

From this point, each leg in the route, starting with the active leg, is inspected. Each leg is updated. For an enroute leg, with a defined starting and ending point, this update does nothing. However, for a more complicated leg type whose end point varies as a function of various flight parameters, this update allows the end point of the leg to be computed. A different computation is performed for each type of leg which may have come from the database. These differing types are as follows:

Once the leg has been updated, its starting and ending points are known. At this point, it can be converted to segments. These segments indicate the path along the ground of the leg, and a given leg may take a different number of segments to represent it. For example, a simple enroute leg may be represented by just one segment, connecting the start and end with a straight line. By contrast, a procedure turn leg may take four segments to complete – A straight path away from the starting fix, a forty five degree turn, a straight path, a one hundred eighty degree turn in the opposite direction, and a straight line to return to the original course. Once this is done, one additional segment is created to connect the leg to the following leg. An illustration of this can be seen in Figure 30 on page 59.

Once these segments are created, each segment is examined with regards to vertical navigation. When first created, the first segment of a leg starts at the same altitude as the start of the leg, and ends at the same altitude as the end of the leg. Any subsequent legs start and end at the same altitude as the end of the leg. Viewed with

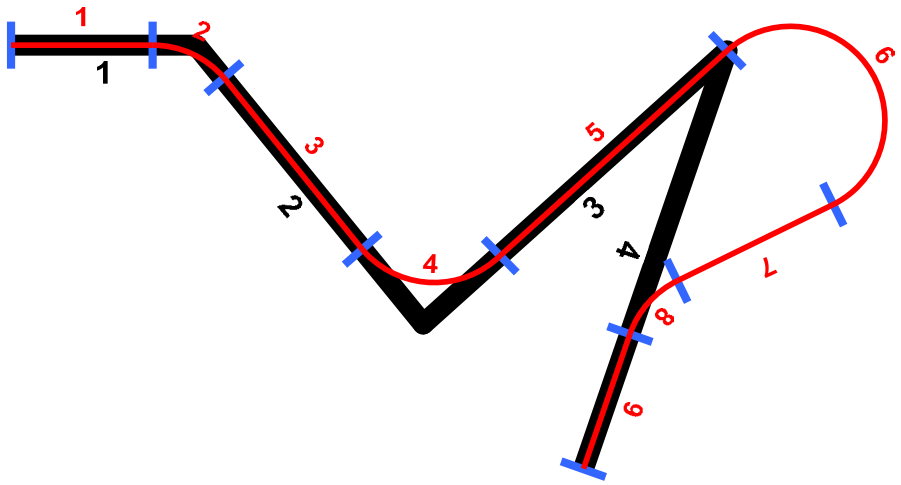


Figure 30 A plan view of a sample route, consisting of five legs

Note: In this diagram, 4 legs, in black are converted to 9 segments, in red. Extra curved segments are added, allowing the route to be flown in a smooth manner. The radius of these curves is the same as the turning radius of the aircraft. Segment 4 fillets between leg 2 and 3. However it is not possible to fillet very sharp turns, such as between legs 3 and 4. In this case, an overshoot and return to path is generated, as can be seen.
Leg 1 constructs segment 1 and 2.
Leg 2 constructs segment 3 and 4.
Leg 3 constructs segments 5 and 6.
Leg 4 constructs segments 7, 8, and 9

regards to vertical navigation, this means that all of the altitude change of the entire leg is assigned to the first segment. This may result in a climb rate or descent rate that is excessive, if the amount of elevation change is large and the length of the first segment is small. To account for this, each segment is examined and corrected with regards to vertical navigation.

Each segment that is created from the legs is examined for elevation change. If an elevation change is required, then the amount of distance that would be required to accomplish the elevation change called for is computed. If the distance required is less than the distance of the leg, then the segment is split into two separate segments at the point where the climb can be completed. This results in the original segment being

represented now as a climb segment and a constant altitude segment. This may be seen illustrated in Figure 31.

By contrast, if the amount of distance required to complete the altitude change is more than the length of the segment, then the amount of change that can be completed in the available distance is computed. The end point of the segment is set to match whatever altitude can be reached. This same altitude is assigned to the start point of the following segment.

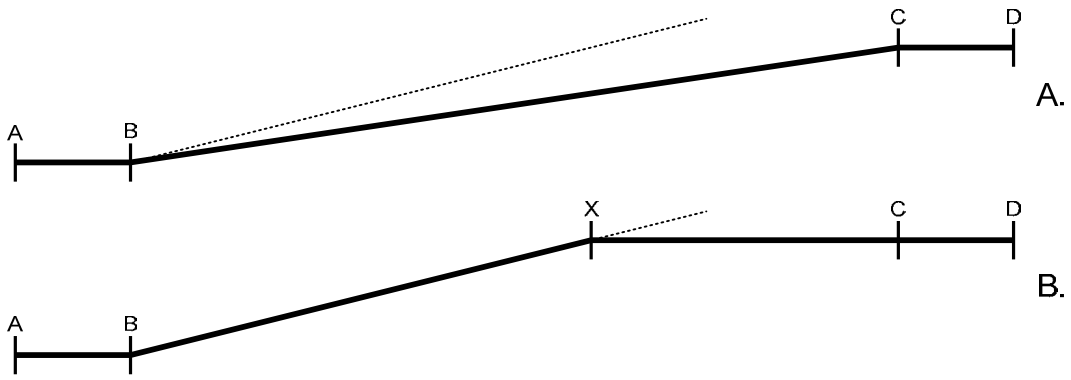


Figure 31 A side view of a series of legs with an altitude change.

In figure A, the route is three segments. The climb segment, BC is shallower than the possible climb angle, seen as the dotted line. In Figure B, after vertical navigation has been computed, The BC segment is split into two segments, BX and XC. Point X is positioned to match with the possible climb rate and the altitude change needed.

The same process is performed on the following segment, and is repeated for each subsequent segment, until the change of altitude can be completed, or the leg ends and the desired altitude has not yet been reached. This step-by-step refinement process is illustrated in Figure 32 on page 61.

Once vertical navigation changes have been performed on each segment for a given leg, the ending point and altitude of the final segment is passed to the next leg. This information is used to update the start of that leg, for use when the leg is updated.

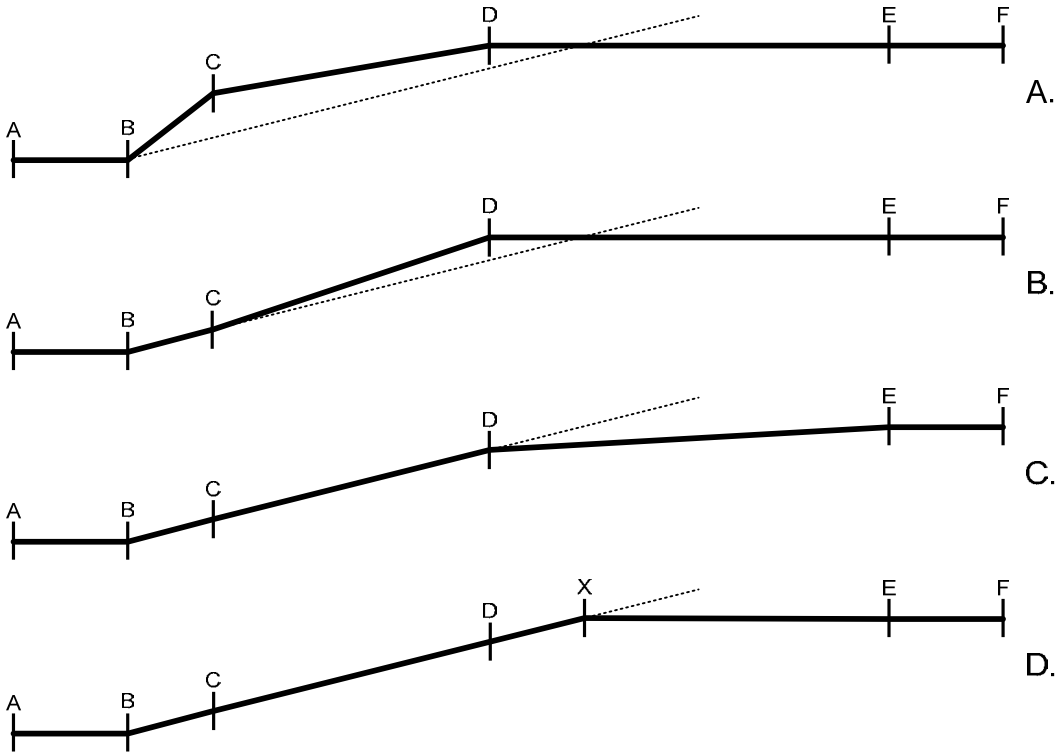


Figure 32 A side view of a second series of legs with an altitude change.

In figure A, a more complicated route before vertical navigation is seen. Two segments, BC, and CD contain climbs. Climb BC is too steep to be possible. Climb CD is shallower than is possible, but starts too high due to beginning at the end of segment BC.
In figure B, segment BC has been processed and is now matched with the possible climb. This leaves segment CD now too steep.
In figure C, segment CD has been processed. There is still climb required, which now appears as a very shallow climb in DE.
In Figure D, segment DE has been processed, by splitting it into two segments, DX, which completes the climb, and XE, which remains flat.

Route Update At Runtime

Once the route is created, a number of operations are performed each frame to update it. First, the segments of the route are examined, to see if the aircraft is being flown along that segment. If so, the aircraft is considered to be on route. In order to be on route, the aircraft must be somewhere between the start and end of the segment. It must be within 2 nautical miles of the centerline of the segment, and it must be traveling

in a direction that is within 30 degrees of the desired direction of travel. If all three of the conditions are met, the segment is considered active. If this is the case, the current leg is found from the segment, and both of these are stored to allow the current leg to be highlighted as it is drawn. Also, certain parameters describing the aircraft's position relative to the route are computed at this time. These parameters are used on the MFD to draw the CDI, and include distance to the left or right of the route, its difference in track from the desired track, and so on.

Servers

A sizable portion of the complexity in the project is involved in collecting relevant terrain and navigation data, in an area around the aircraft, to be shown on the various displays handled by the main thread. There are four types of data which must be accessed in this way, with very different content.

Terrain Data

The first is terrain data, which is a large collection of elevation points in an evenly spaced grid. Terrain data is stored into files of one degree of latitude by one degree of longitude. These files are stored at a resolution of six arc seconds, or a spacing of approximately 200 meters, between each post. The terrain data is loaded by the application in rectangular cells of one tenth of a degree by one tenth of a degree, in other words, an area of one one-hundredth of the source file. Indexing of this data for loading is simple. Given the position of the aircraft, it is easy to compute latitude and longitude of nearby cells. Each of these is rounded to the nearest integer degree to determine the name of the file which contains that cell. Once the file is found, it is segmented and the appropriate section is loaded into memory. This process is not particularly resource intensive; however a single file cannot be processed in the amount of time devoted to a single frame of the display app. For this reason, it is necessary to handle terrain in a

background thread to be loaded asynchronously to the main thread. Once the cell is loaded, it is necessary to pass it to the main thread in a thread safe way.

Arena Storage

For this, a class called an arena is used. Structurally, the arena consists of two lists, one for newly added cells, and one for the main list which appears on the screen. The added cell list is protected by a mutex. The structure and interface of the arena can be seen in Figure 33.

Each time the background thread completes loading a cell, it attempts to lock the mutex on the add list. If it succeeds, the cell is placed into the add list, the background thread releases the mutex, and then carries on with loading the next cell. Asynchronously to this, the main thread periodically checks the list. It does this by locking the mutex on

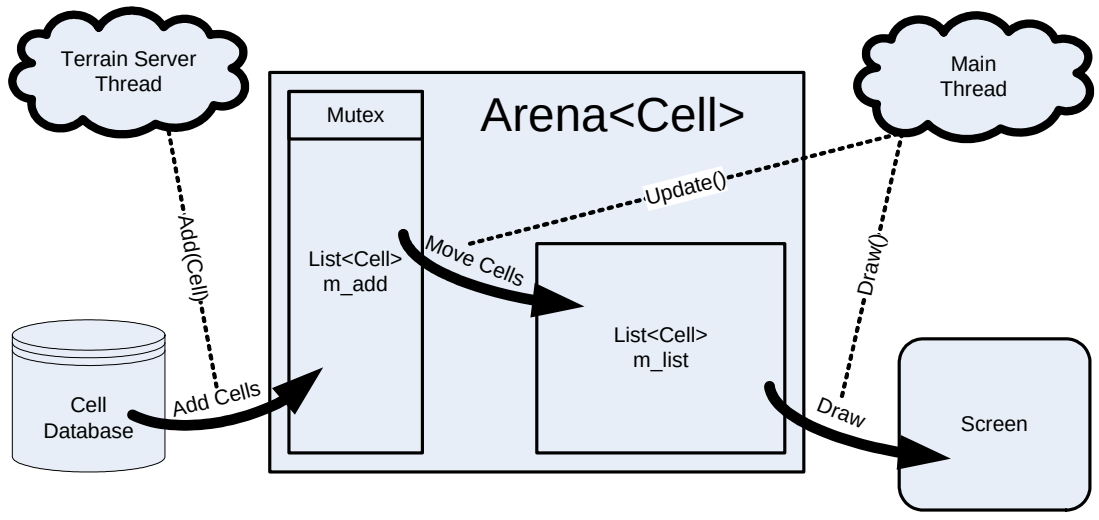


Figure 33 A depiction of an arena of cells.

Note: In this figure, the cell is the basic unit of terrain data. The terrain server thread reads cells from the database, and adds them to the arena. Within the arena, the sells are stored in the list of added cells. The Main thread periodically calls update, which moves cells from the added list to the main list. These two actions are locked with a mutex, so that only one of the two may modify the added cell list at any given time. Once the cells are in the main list, they may be drawn at any time without fear of thread safety issues.

the add list, and examining the add list. If the list has no items stored in it to be handed over, the mutex is released and the main thread carries on. If not, the mutex is held, a fixed number of items are pulled from the add list and processed into the main list, and the mutex is released. It is necessary to take only a fixed number to avoid slowing down the thread if a large collection of objects come into the queue at the same time. Limiting the number processed allows the time needed to import the data to be amortized across several frames. It is possible for the two threads to compete for access to the queue. In this case, the mutex arbitrates access. If the background thread has locked the mutex when the main thread attempts to access it, there is only a brief delay, as adding a block to the queue is a very quick operation. However, if the main thread has locked the queue, the background thread will wait until the main thread has completed taking all items it wishes to from the queue. This is a slightly longer delay, but is no major concern, as it only causes the background thread to wait.

Navigation Data

The next class of data which is served to the main thread is navigational data. Navigation data is taken from the source data in either DAFIF or ARINC 424 format. These two formats are similar, consisting of fixed-length records, with each record representing a particular object, such as a VOR transmitter or a runway endpoint. This is an appropriate format for transferring data as it is a clear, system independent text format.

It is not, however, particularly well suited to fast access or searching. For this reason, an external application is used to convert the data from the text format to a relational database. This database is then stored and accessed by the main application. When loaded, the data is accessed via a standard SQL interface. This allows for simple, human readable database queries to be created. These queries are then passed to the database engine, which retrieves results matching the request. Searching of the database tables for records matching a specific query may take several seconds. For this reason,

the searching is again done in a background thread, and the results are added to a queue when they have been processed. The interactions of the background thread, main thread, and queue are identical to the case of terrain data.

Requested Data

Both terrain and navigation data are handled by a background thread which is responsible for ensuring that the main thread is provided with data that is near to the aircraft, thus ensuring that an appropriate display is shown on the screen. By contrast, when creating a route, for example, the pilot may need information about a navigation feature which is very far from the aircraft, perhaps hundreds of miles away. To handle this case, a third background thread exists to manage data demanded by the main thread. In this case, a queue is again used, but the data transfer is in the opposite direction. The main thread creates a data structure representing a request for data. This request is then placed into a queue to be delivered to the background thread. Once passed as described above, the background thread examines the request, and finds results in the database which match, storing them into fields in the request data structure. Once it has finished with the request completely, it sets a flag, marking the request as complete. The main thread continues to watch the request structure until it notices the flag marking it as complete. At this point, it examines the request and uses the data as needed. In this case, it is not necessary to use a mutex, as the two threads are not competing for access to the data. Instead, the main thread is simply waiting for the background thread to indicate that it has completed its processing. This can be done with a simple variable, rather than a full mutex.

Weather Data

Finally, a fourth background thread handles weather data from the NEXRAD weather receiver. In this case, the background thread connects to the receiver via a USB connection, and requests certain data be sent on a periodic basis. Once that has been

done, the thread listens to the various data sent by the receiver, as it arrives. This happens on no specific timeframe, only as the receiver chooses to forward data. When the weather thread notices that the weather receiver has sent a valid update to the weather data, it processes it into a format suitable for use on the display, divides it into manageable blocks, and passes all of these blocks, in a similar way to terrain data, into a dedicated weather arena.

SYMBOLOGY AND FUNCTIONS

Individual portions of the Synthetic Flight Bag display have been discussed with regard to the software behind the implementation of these displays. It is also necessary to illustrate the presentation and use of these displays and menus.

Symbology

The Synthetic flight bag main display and its major components may be seen in Figure 34. Subsequent figures illustrate the symbology of each component of the display.

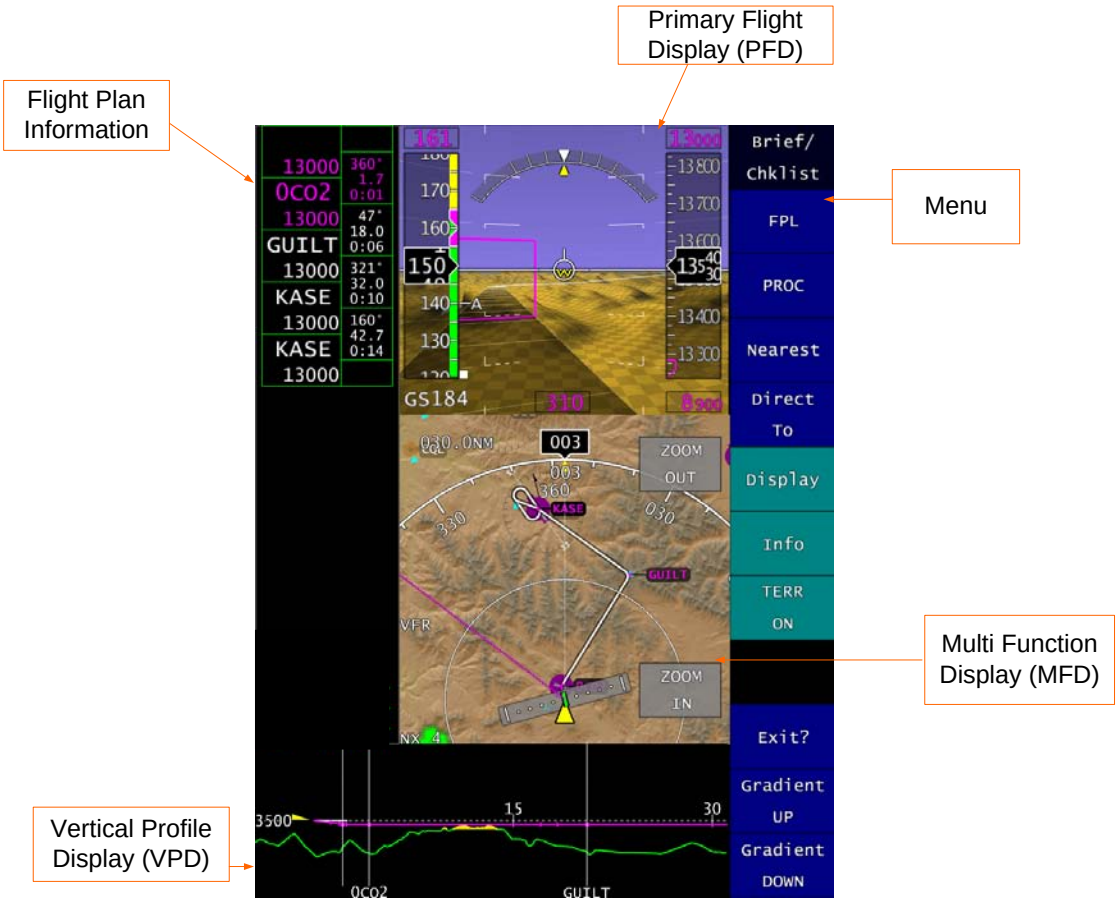


Figure 34 The main screen of the Synthetic Flight Bag.

Note: The display consists of a combination of a PFD and MFD, displayed in the center of the screen. Below these, a VPD is shown. To the left, a flight plan information window is displayed. To the right is the menu.

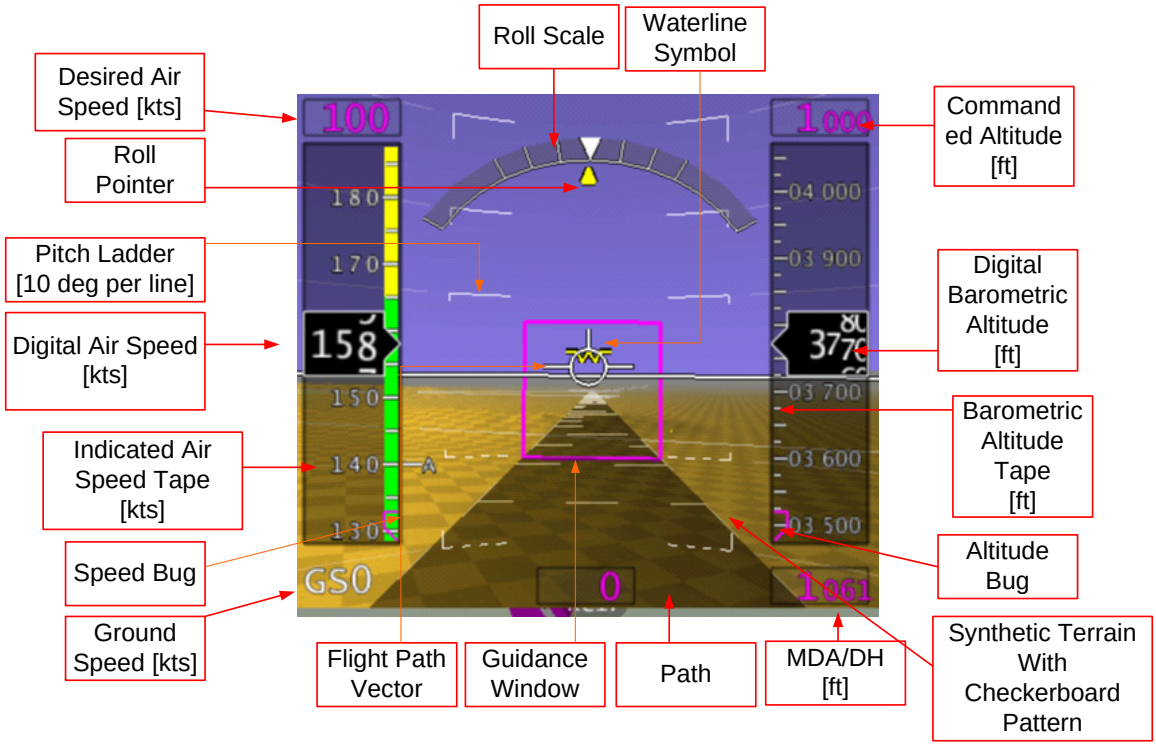


Figure 35 The Primary Flight Display.

Note: Within the PFD, a great variety of information is available. The most prominent feature is the attitude indicator, showing blue sky over brown terrain.. This display moves to indicate the pitch and roll of the aircraft. A pathway, seen as the brown ribbon at the bottom of the screen, guides the pilot along their flight plan. The overall guidance concept is that the flight path vector is to be placed in the center of the magenta guidance window. As is standard among glass cockpit displays, airspeed is seen at left, and altitude at right. The airspeed tape is color coded, matching with the standard arcs seen on a traditional round airspeed indicator. The magenta colored displays (Desired airspeed, commanded altitude, MDA/DH) may be touched. When this occurs, a keypad on the MFD may be used to enter a new command in these windows. These commands are displayed in the command buttons, and also set the position of the bugs on the airspeed and altitude tapes.

Source: SFB Pilot Guide

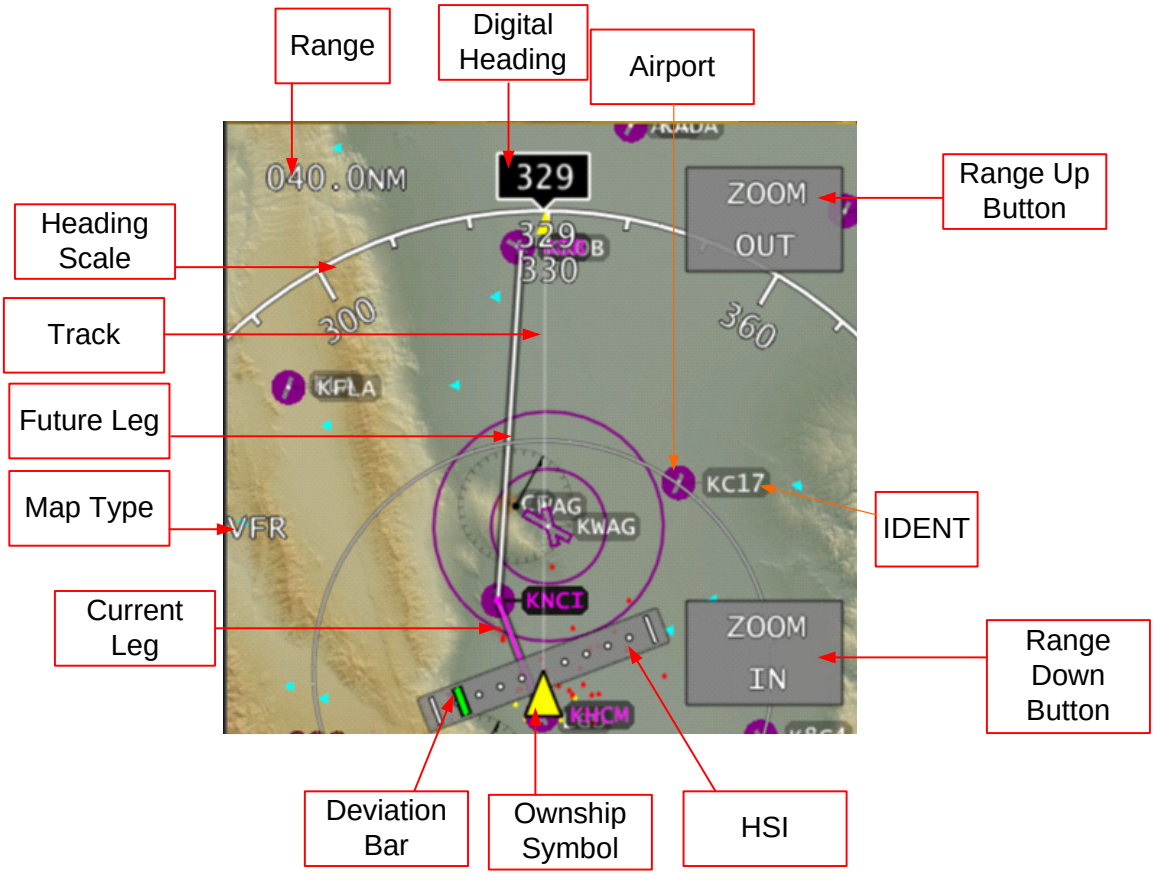


Figure 36 The Multi Function Display.

Note: The MFD, displays a moving map to the pilot, and includes a view of terrain, nearby obstacles, the current path, navaids, and weather. It also provides a CDI, to display more accurate guidance information to the pilot when they are far from their path. A heading display and compass rose allow the pilot to accurately fly headings, as is often required during maneuvering under the control of ATC. Following standard conventions, the current leg of a route is drawn in magenta, while future legs are drawn in white. Airspace boundaries are also shown, increasingly important in modern aviation, where blundering into a controlled airspace may have serious consequences. Seen below, the airspace around KWAG may be seen as a pair of concentric circles.

Source: SFB Pilot Guide

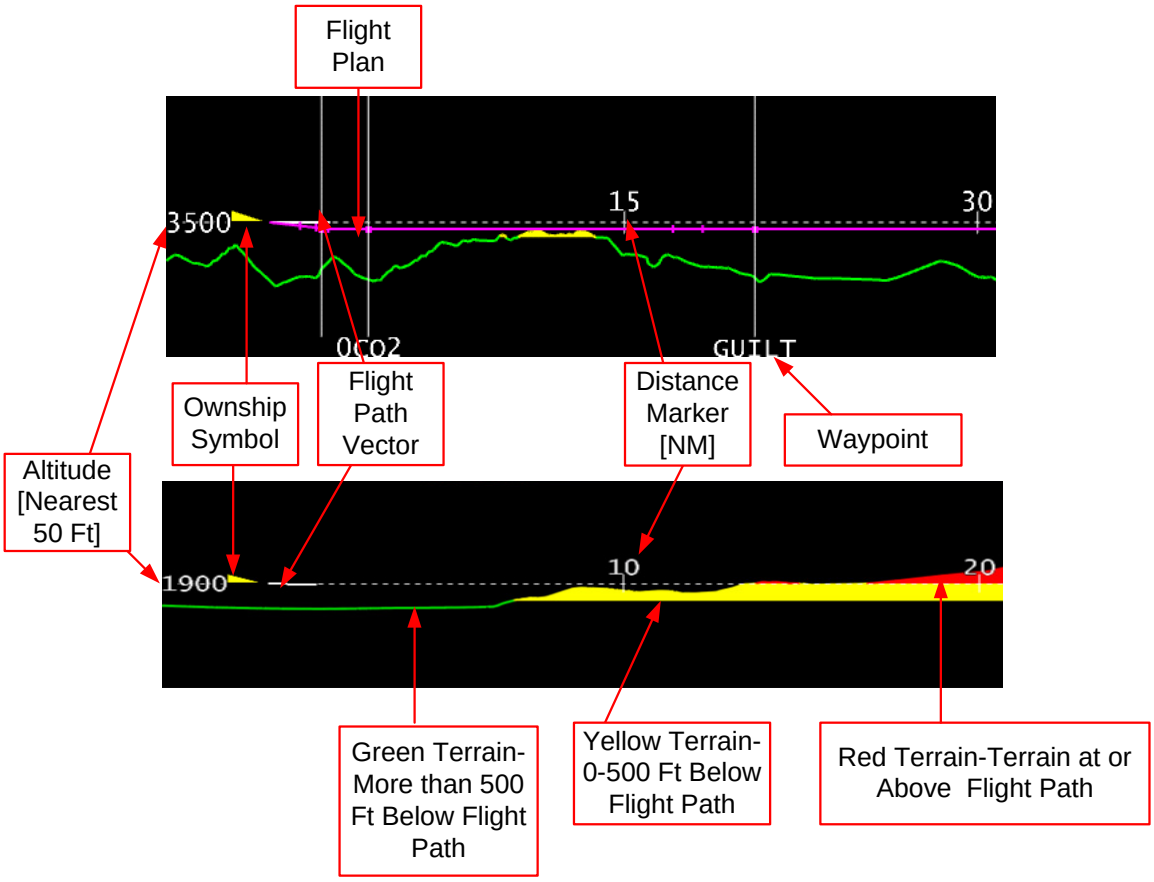


Figure 37 The Vertical Profile Display

Note: The Vertical Profile Display illustrates a side view of the terrain in the direction of the aircraft's flight. It also presents a depiction of the route, allowing the pilot to judge whether altitudes chosen for future segments are appropriate for terrain avoidance. This view also allows for good estimation of distance required to complete a climb, as well as distance to terrain features in front of the aircraft.

Source: SFB Pilot Guide

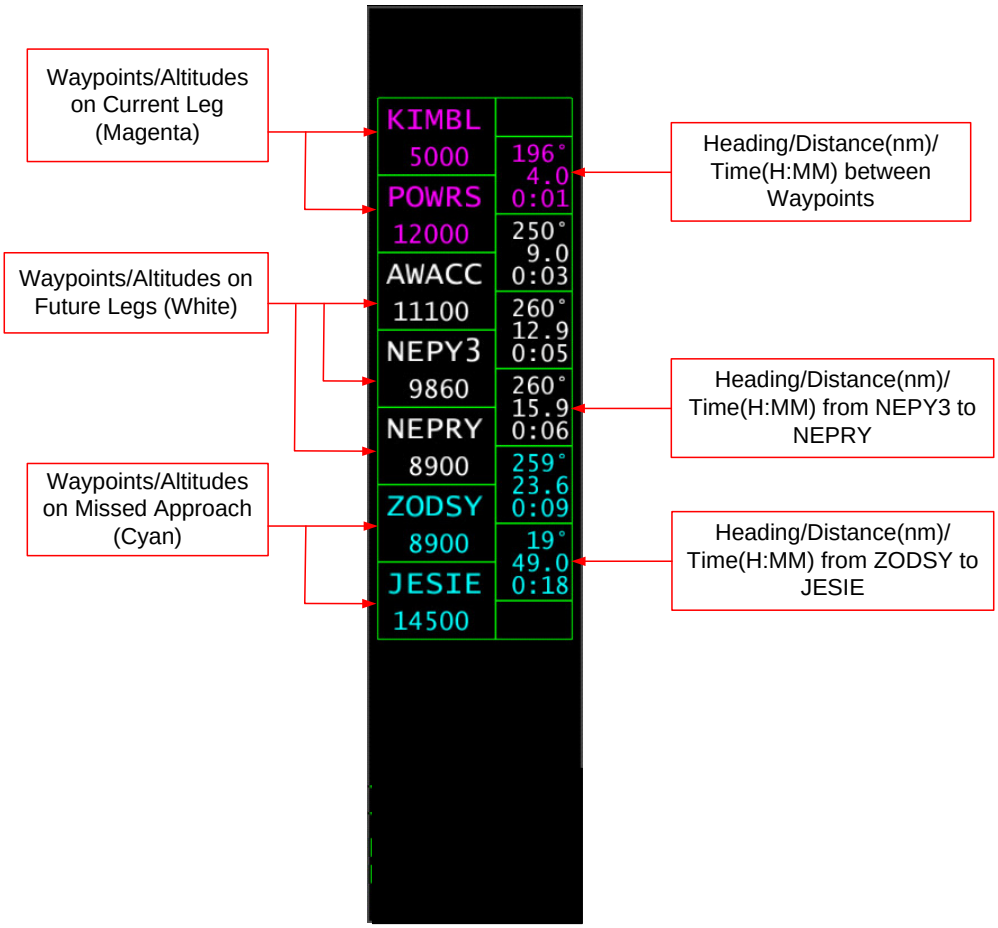


Figure 38 The Flight Plan Information Window

Note: The flight plan information window is used to convey information to the pilot about their currently entered route. Information about each leg is indicated, by display of a starting and ending fix for each leg, each with altitudes. Also shown is the course, distance, and time between the two waypoints. The active leg, if one exists, is drawn in magenta. If a missed approach is included on the flight plan, the missed approach point and all subsequent legs are drawn in cyan.

Source: SFB Pilot Guide

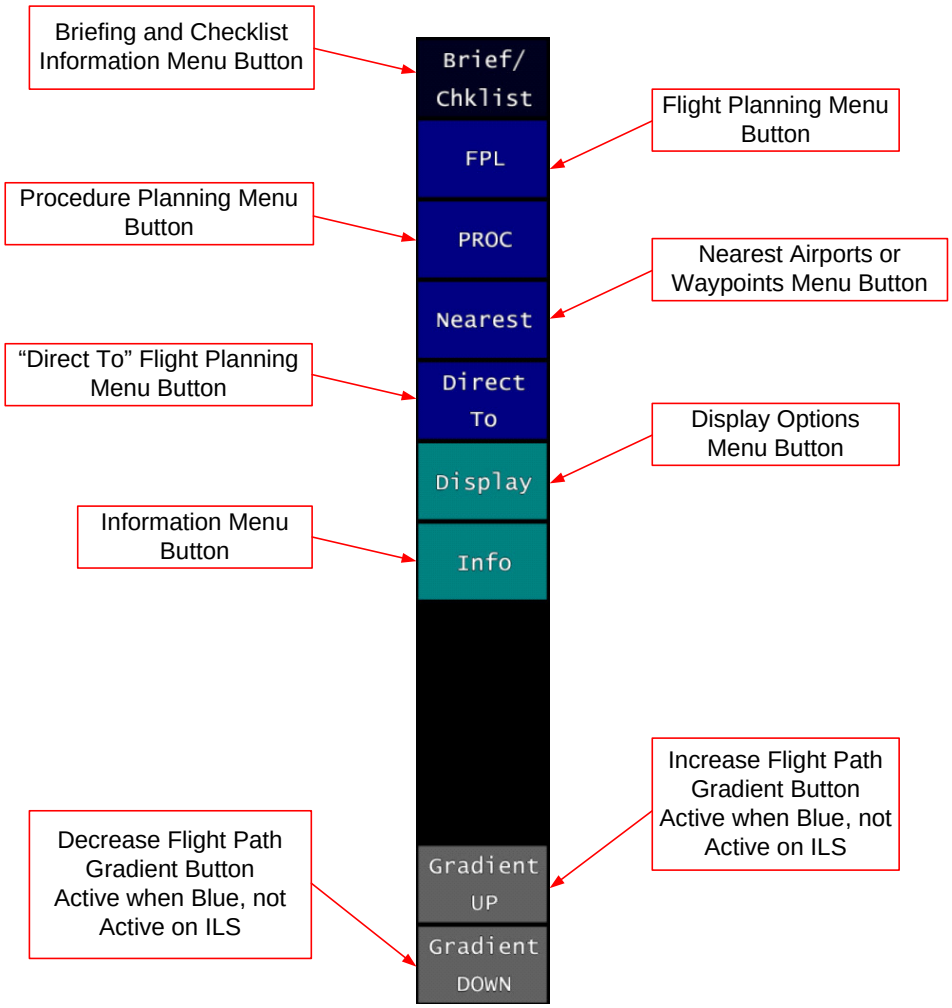


Figure 39 The Main Menu

Note: The main menu provides a means to enter the various submenus. Abbreviations such as FPL and PROC are standard abbreviations in aviation.

The gradient up and gradient down buttons are active during climbs and descents and are used to adjust the angle of climb or descent of the route to match the aircraft performance. These two buttons are placed on the main menu due to frequency of use and importance.

Source: SFB Pilot Guide

Functions

The following diagrams and notes are taken from the Synthetic Flight Bag Pilot's guide, and were used in the flight test to illustrate the features of the display to pilots involved in the study.

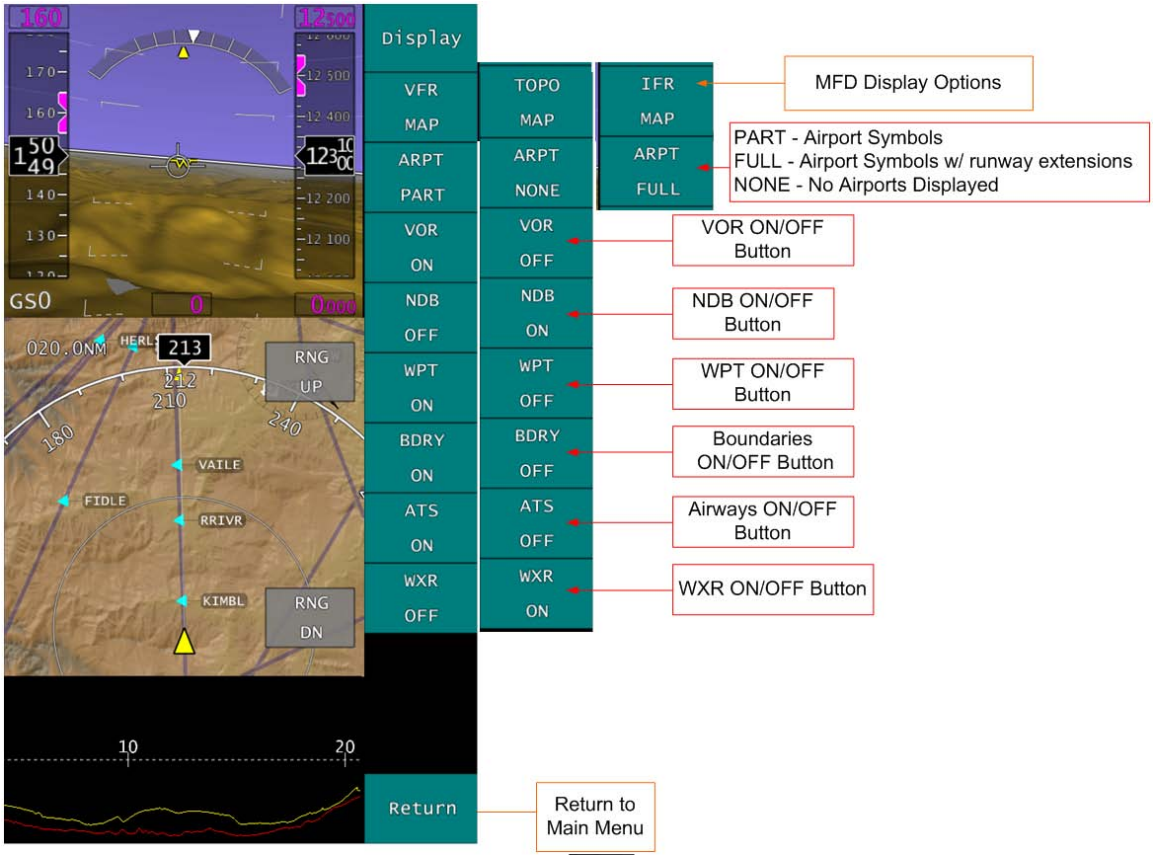


Figure 40 Display Options

Note: Buttons on the display menu indicate the current state of the MFD. For example, when waypoints are shown, the waypoint button label will indicate "WPT ON". Touching the buttons will cause them to cycle through the possible options.

Source: SFB Pilot's Guide

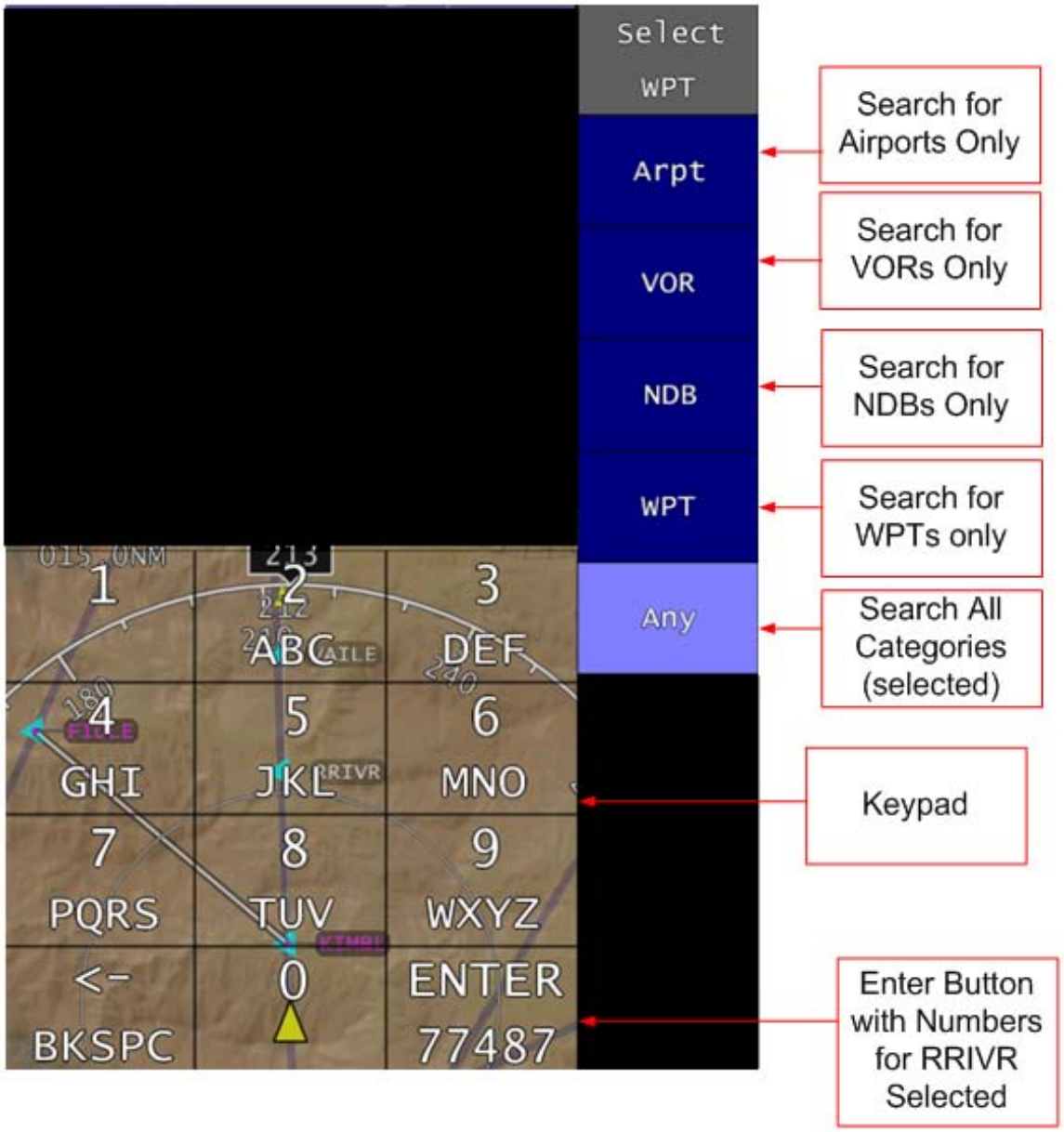


Figure 41 The Search Menu

Note: The search menu is used to search for WPTs and to enter information
To use the search menu, begin by selecting the type or types of WPT to search for. Next enter numbers corresponding to the letters of the WPT being searched for, then use the enter key to start the search
A distance sorted list of possible results are then displayed and the correct WPT can be selected from the list.

Source: SFB Pilot's Guide

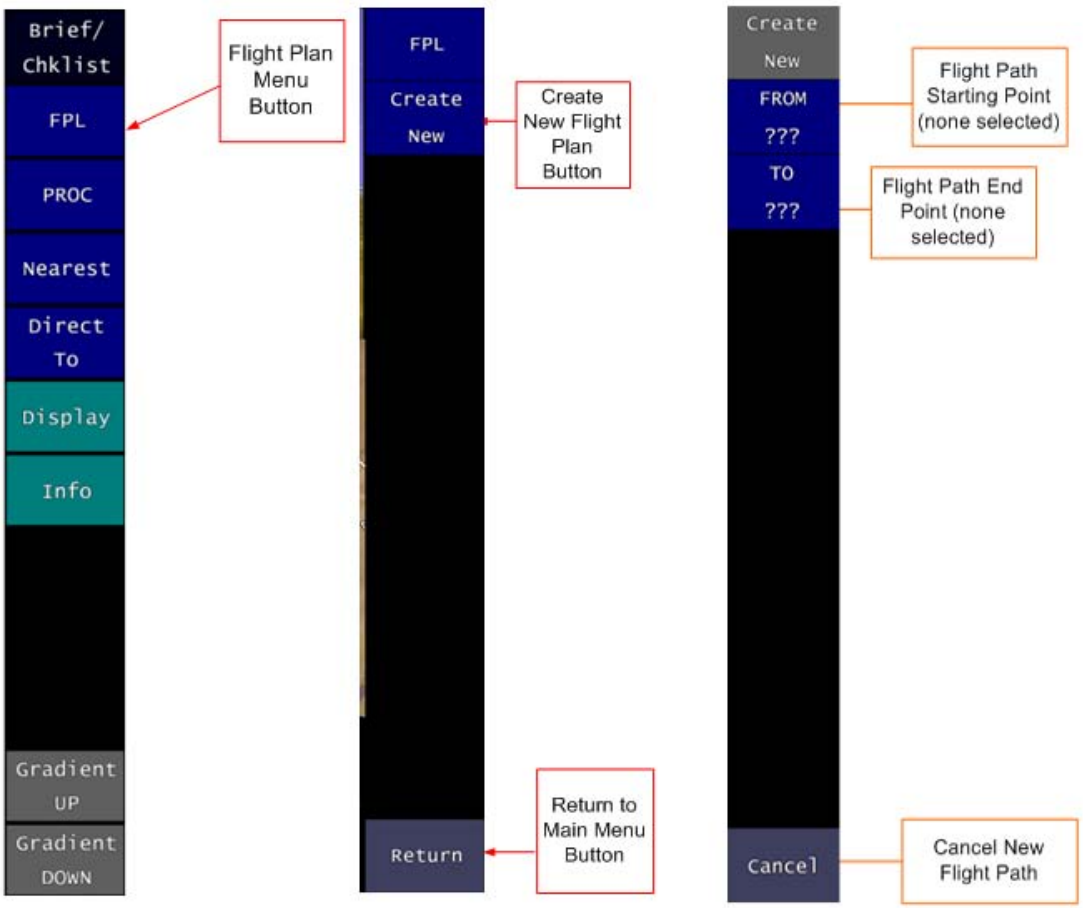


Figure 42 Creating a New Flight Plan

Note: To create a new flight plan
Select the flight planning menu button on the main menu
In the flight planning menu select Create New
To set start and end points, select either the FROM (starting point) or TO (end point) buttons in the create new menu

Source: SFB Pilot's Guide

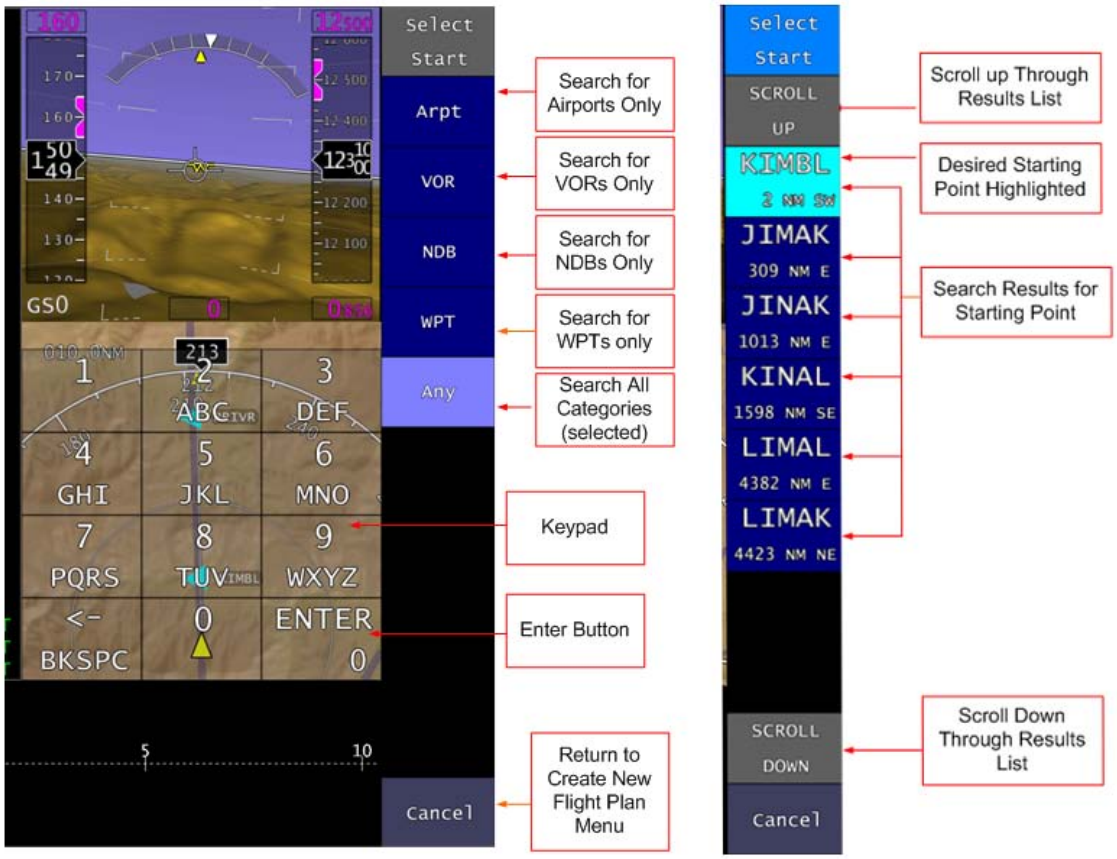


Figure 43 Selecting a Starting Point

Note: After selecting either FROM in the create new menu, select the type(s) of WPT to search for
Select the numbers corresponding to the letters of the WPT being searched for, then select ENTER
In the search results column, highlight the correct WPT and confirm the selection using the Select Start button

Source: SFB Pilot's Guide

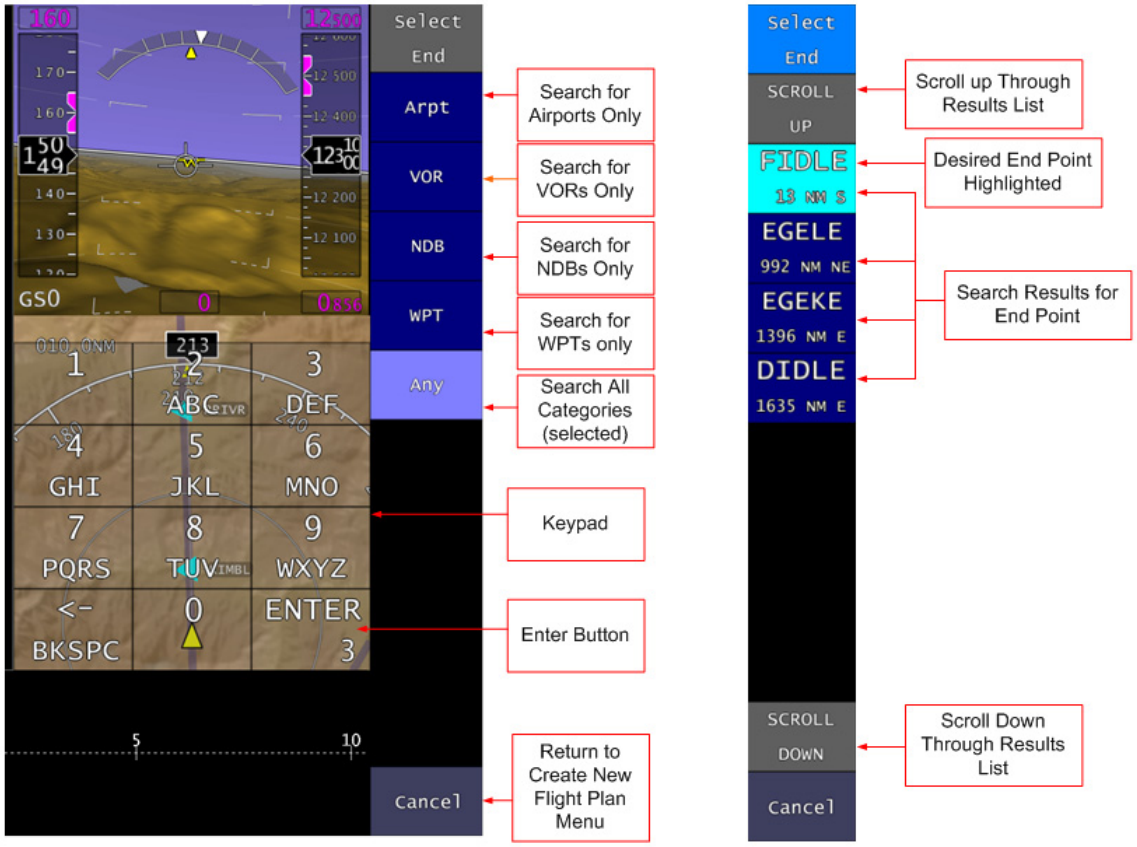


Figure 44 Selecting an Ending Point

Note: To choose an end WPT, select TO in the create new menu
The end WPT is selected the same way as the start WPT by using the keypad and then selecting the correct WPT from the results column

Source: SFB Pilot's Guide

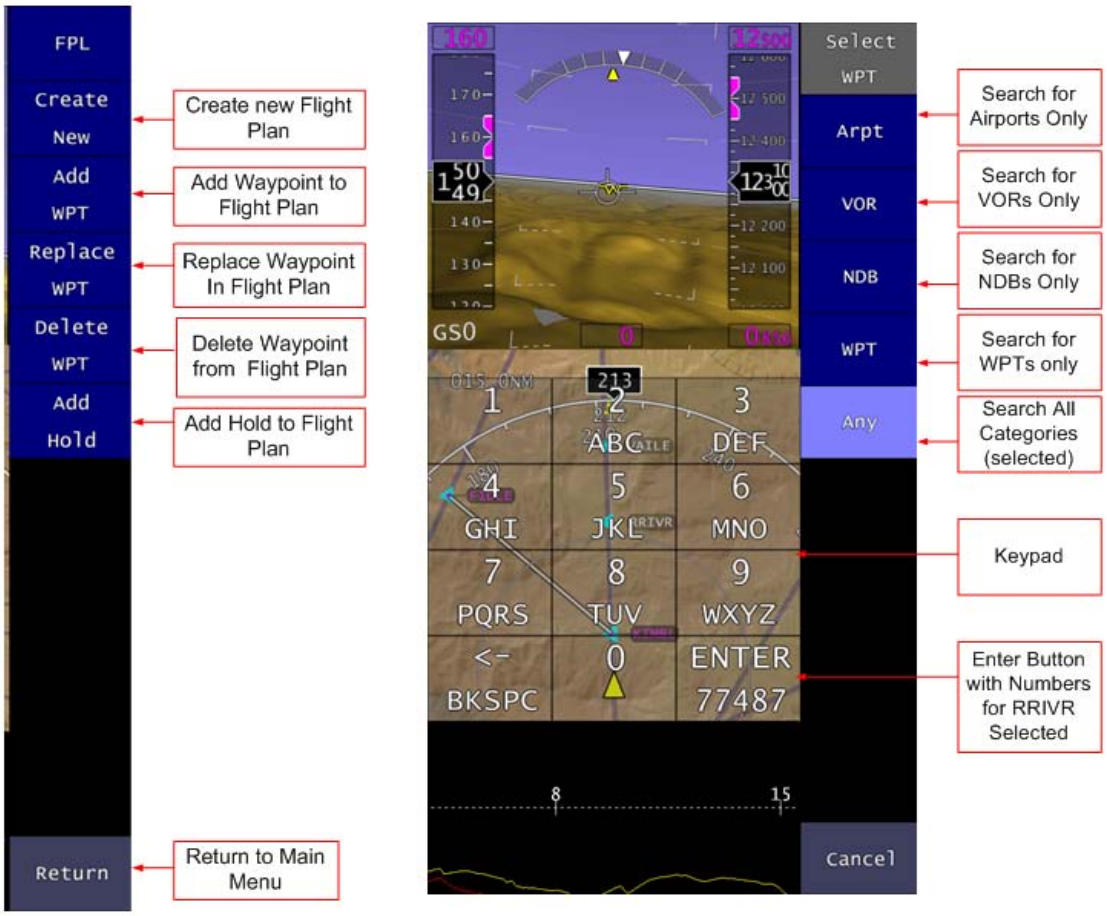


Figure 45 Adding a Waypoint to a Flight Plan

Note: With a flight plan already entered, select Add WPT from the flight plan menu
In the Select WPT menu, select the category of points to be searched
Use the keypad to enter the numbers corresponding to the letters of the point

Source: SFB Pilot's Guide

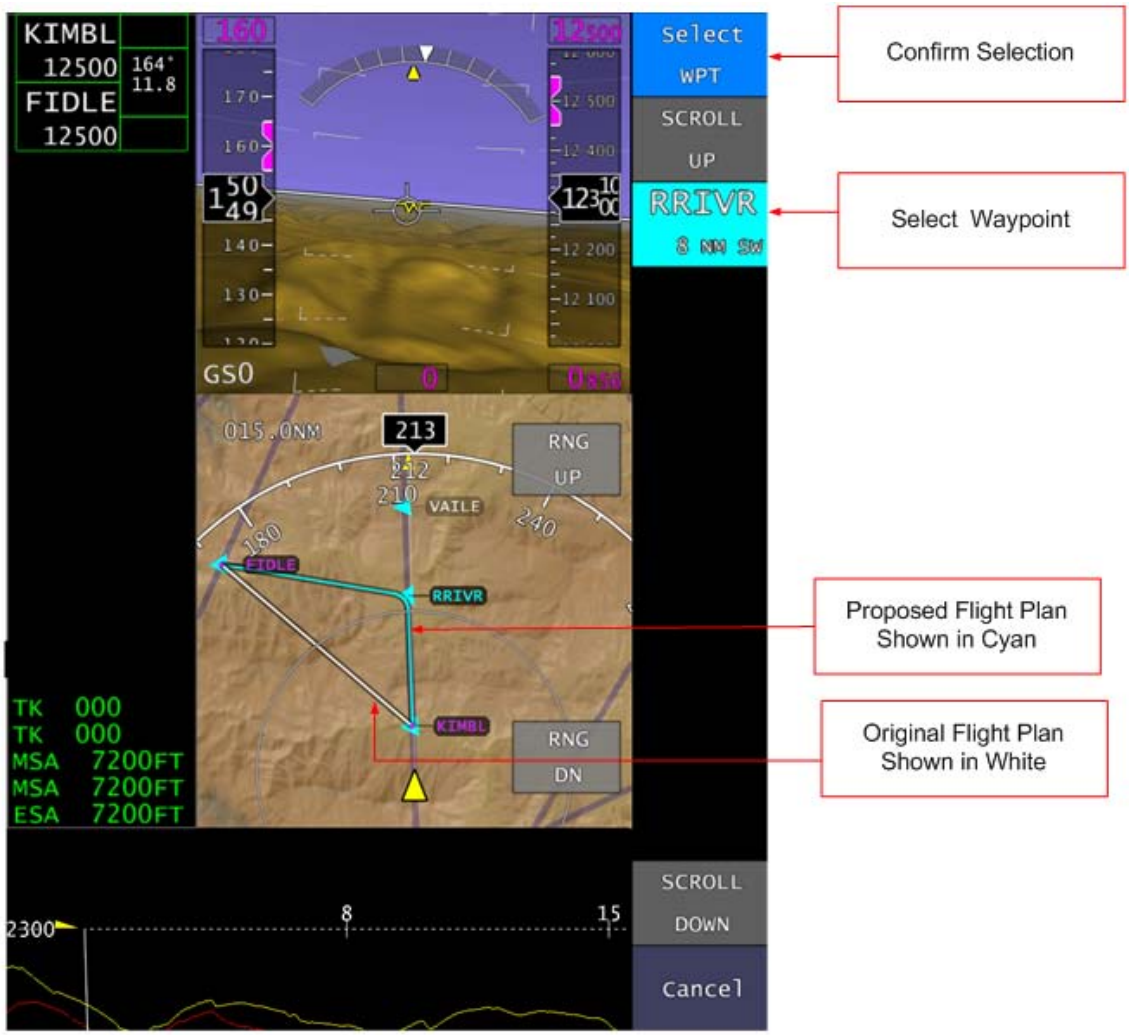


Figure 46 Adding a Waypoint to a Flight Plan

Note: In the search results list, highlight the desired waypoint. The proposed flight plan will appear in cyan on the MFD.
Confirm the selection with the Select WPT button

Source: SFB Pilot's Guide

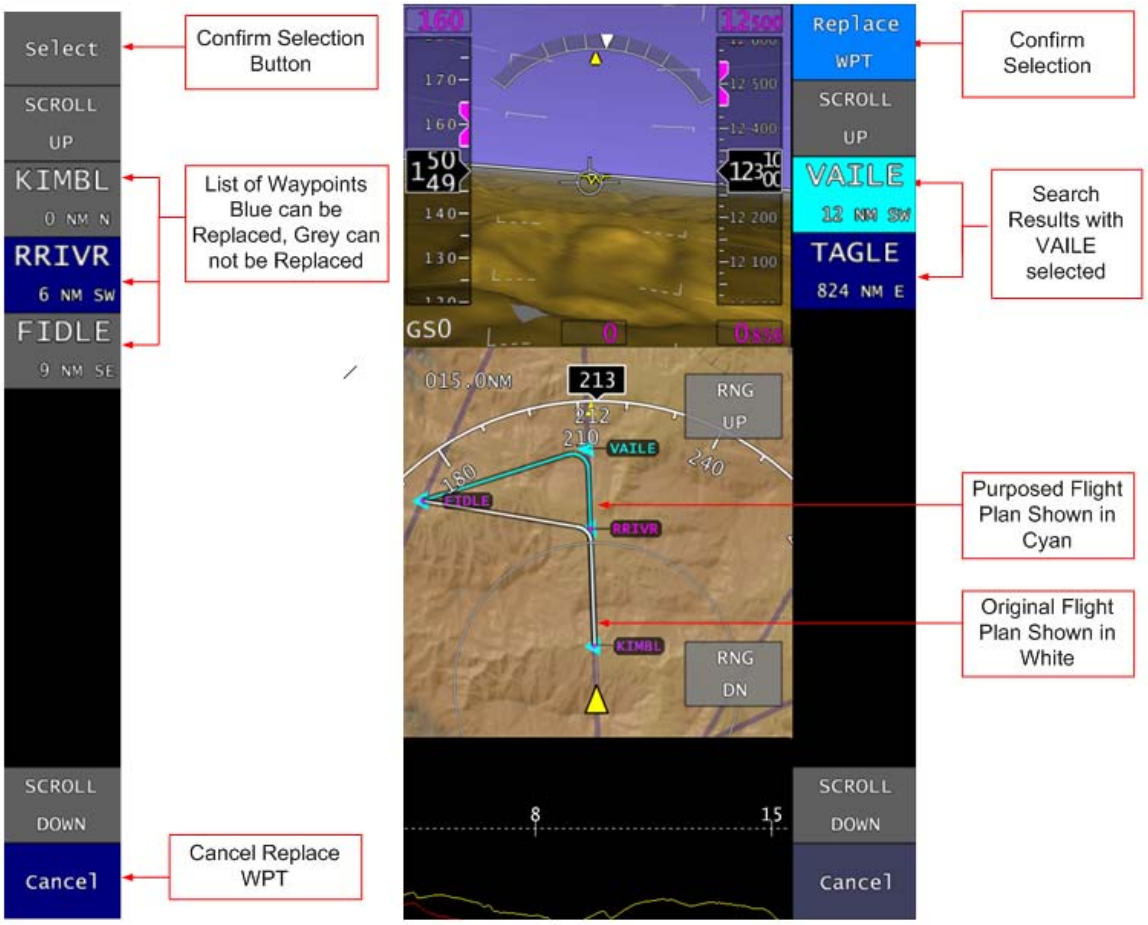


Figure 47 Replacing a Waypoint

Note: Select Replace WPT in the flight plan menu
In the Replace WPT menu select the waypoint to be replaced. Only waypoints shown in blue can be replaced. Confirm the selection with the Select button
Use keypad and search options to search for replacement Waypoint
In results list, highlight correct point and confirm using Replace WPT button

Source: SFB Pilot's Guide

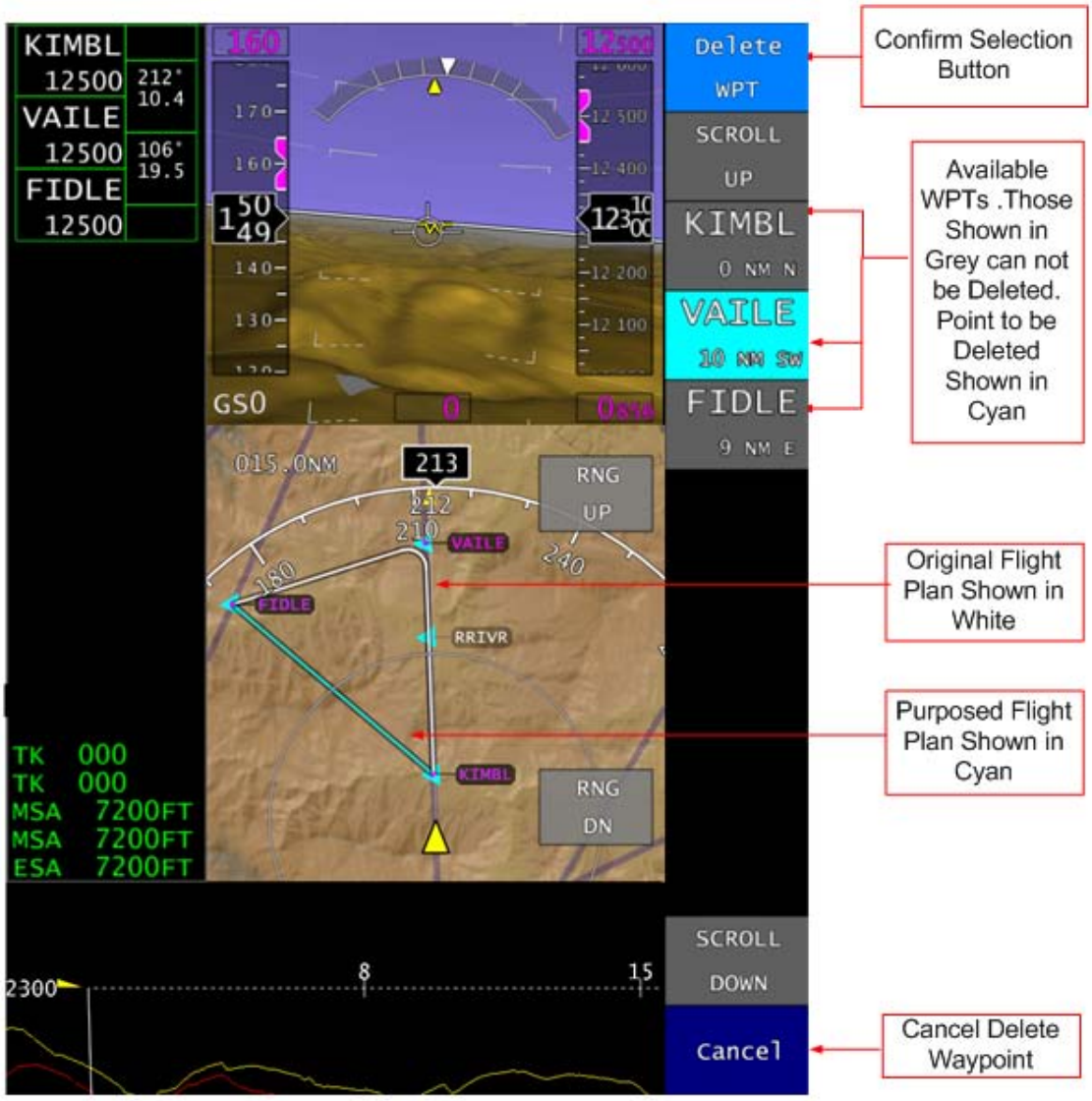


Figure 48 Deleting a Waypoint

Note: In the flight plan menu, select the Delete WPT button
Waypoints that can be deleted are shown in blue. Highlight the point to be deleted. The purposed flight plan appears in cyan on the MFD.
Confirm selection using the Delete WPT button

Source: SFB Pilot's Guide

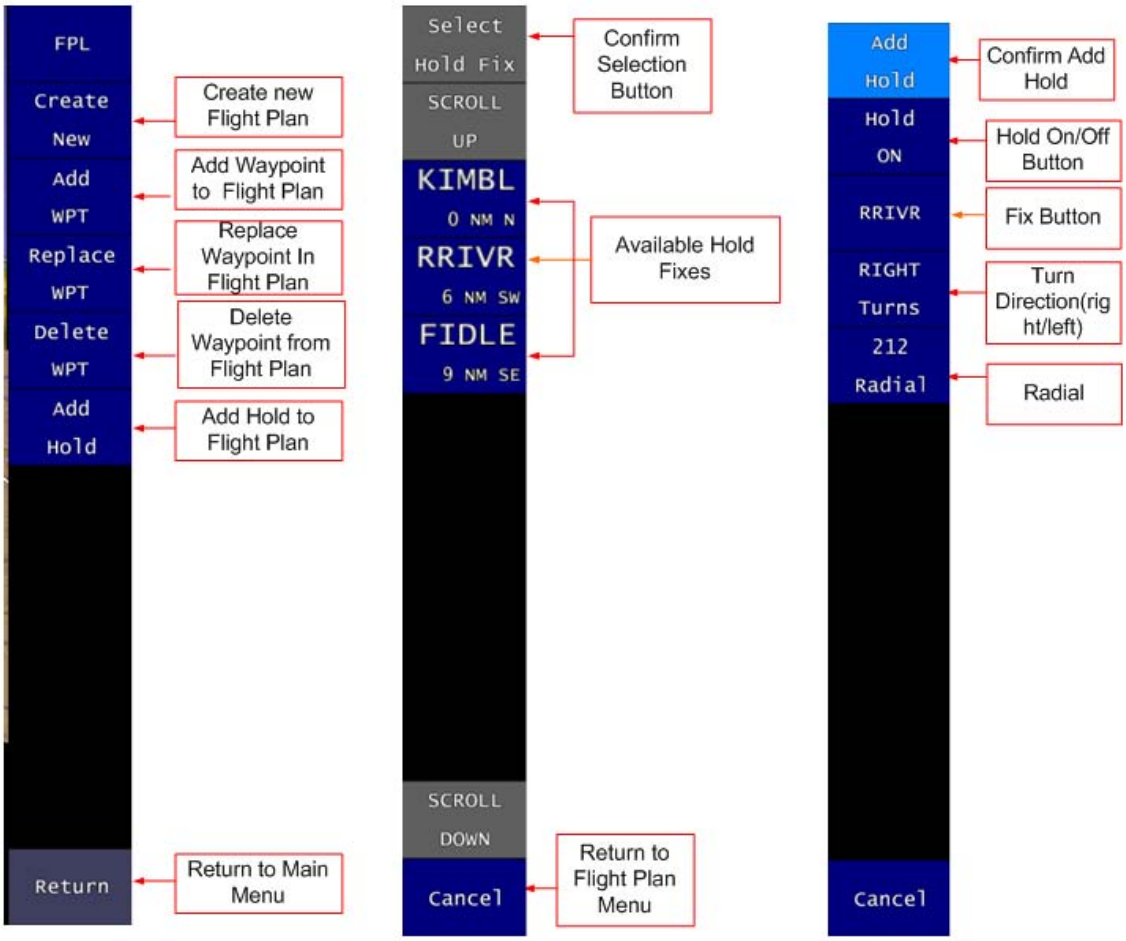


Figure 49 Adding a Hold

Note: In the flight plan menu, select Add Hold
In the Select Hold Fix menu, highlight the correct fix and confirm selection with the Select Hold Fix Button
In the Add Hold menu, check that all settings are correct. The fix can be changed using the fix button, and right or left turns can be selected using the turn direction button

Source: SFB Pilot's Guide

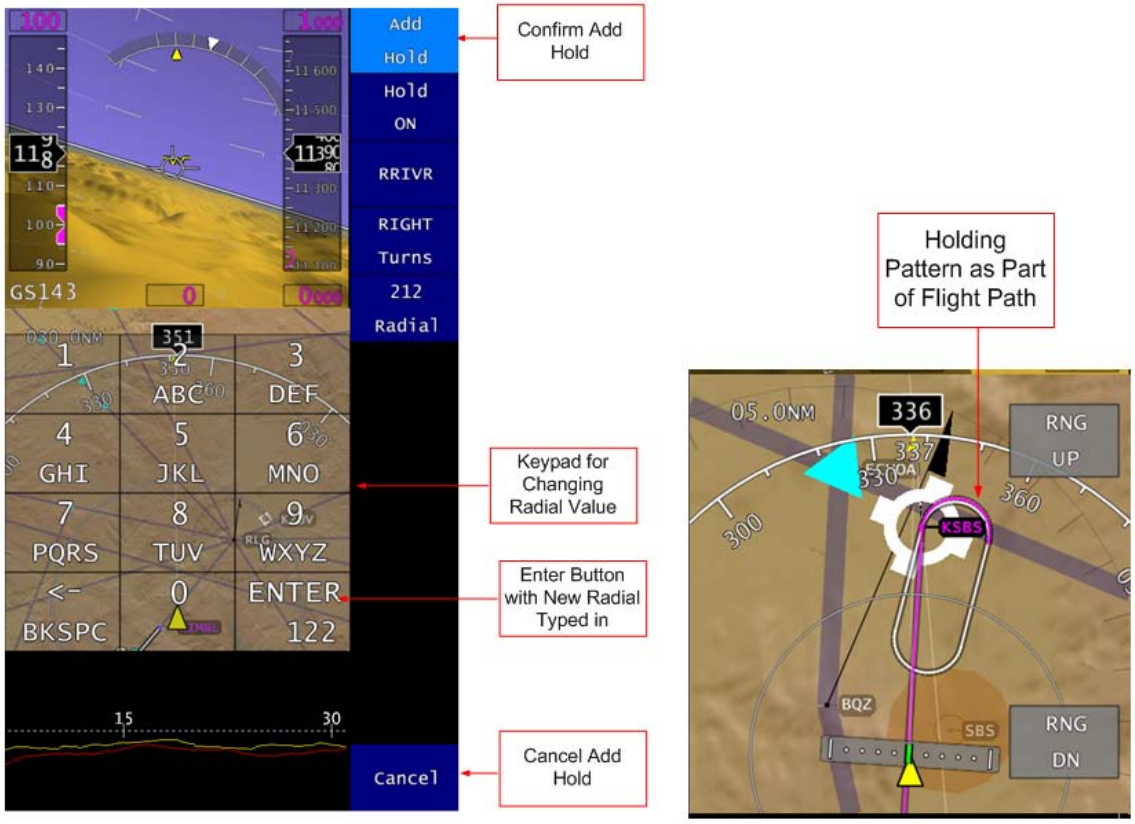


Figure 50 Adding a Hold

Note: To change the radial, select the Radial button. Use the keypad to input the new radial value and use the ENTER button to confirm the selection.
Confirm the hold using the Add Hold button. The holding pattern appears on the flight path on the MFD

Source: SFB Pilot's Guide

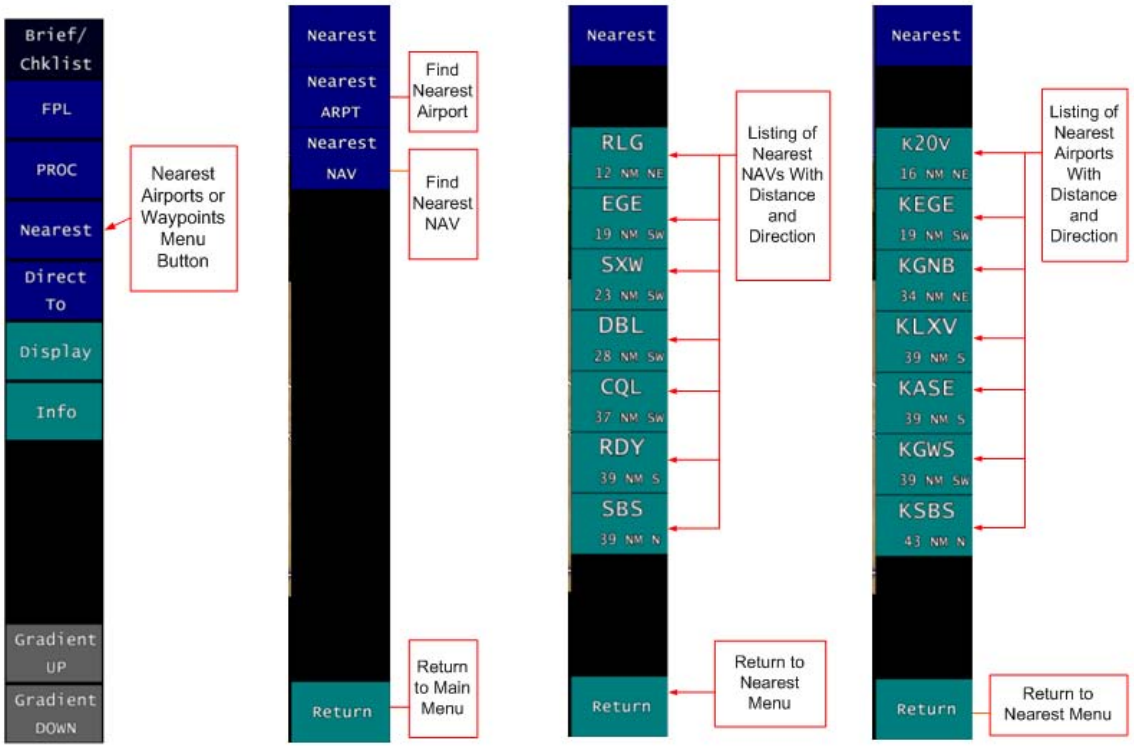


Figure 51 Finding Nearest ARPTs and NAVs

Note: On main menu select Nearest
To find nearest airport, select Nearest ARPT in Nearest menu. Highlight airport on results list and confirm choice using Nearest button
To find nearest NAV, select Nearest NAV button in Nearest menu. Highlight NAV in the results list and confirm using Nearest button

Source: SFB Pilot's Guide

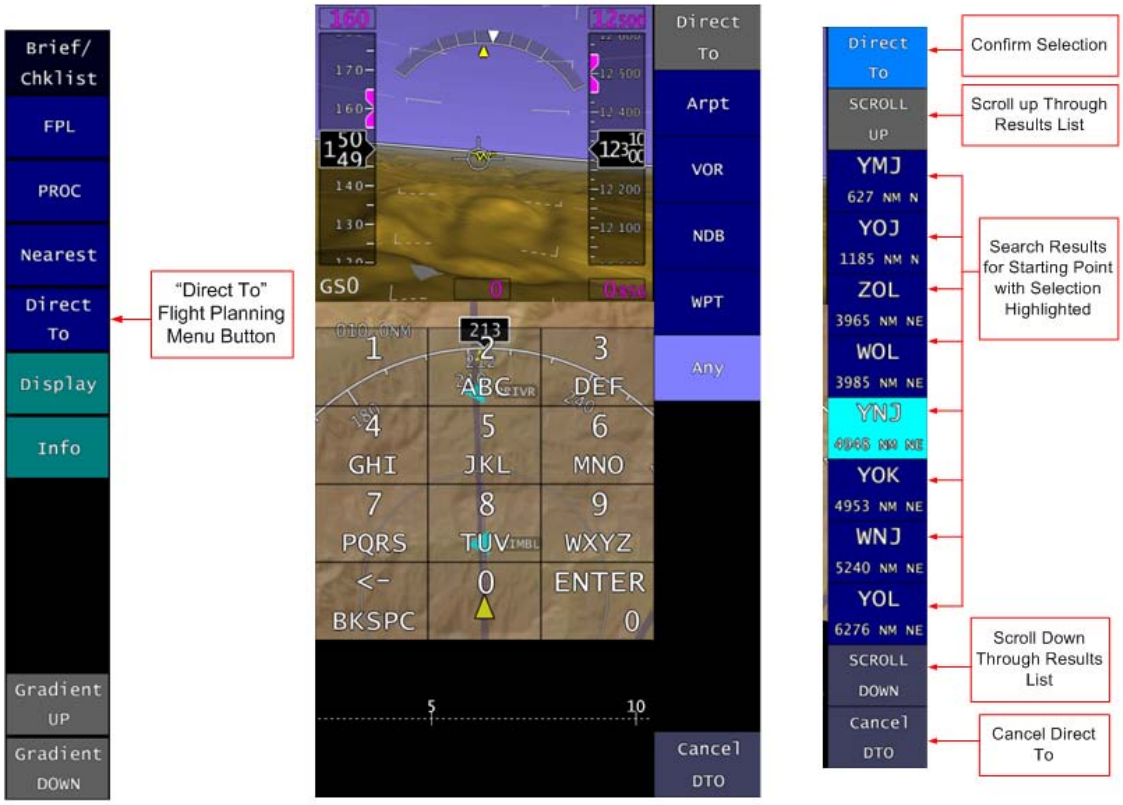


Figure 52 Using Direct to

Note: On main menu, select Direct To
Use search options and keypad to search for WPT
On results list, select correct point and use the Direct To button to confirm selection

Source: SFB Pilot's Guide

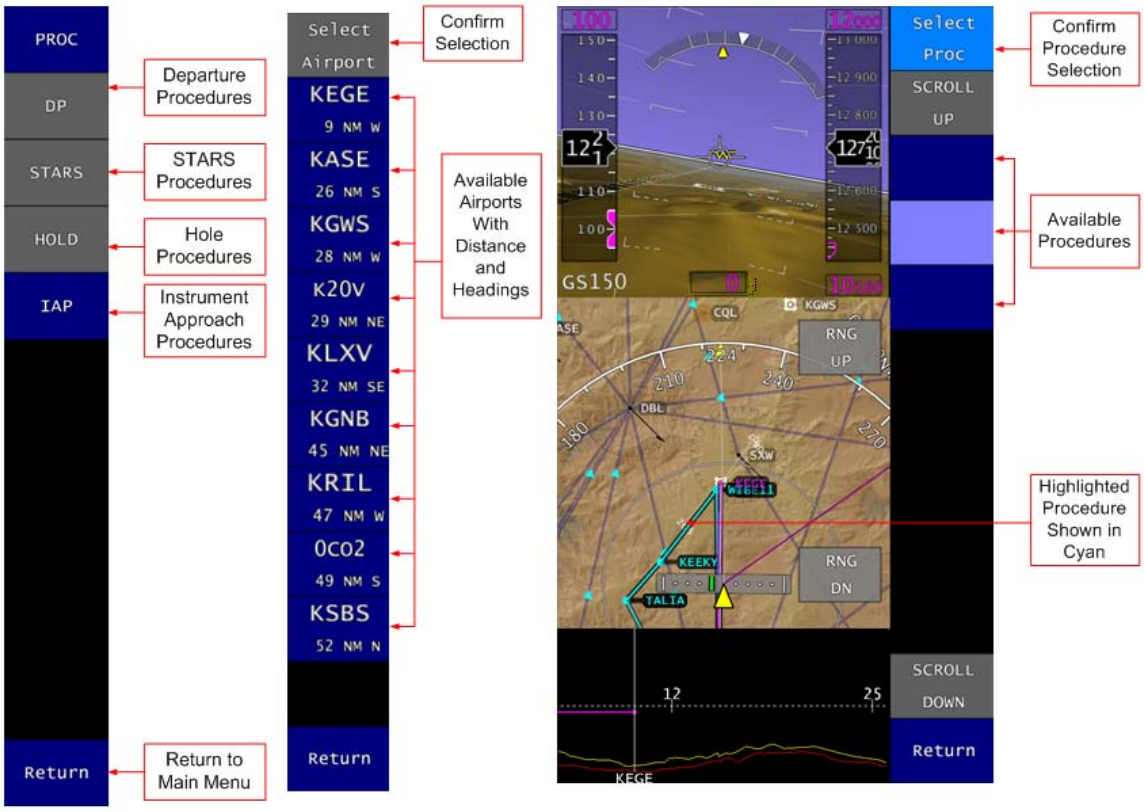


Figure 53 Selecting an Instrument Approach Procedure

Note: In the main menu select PROC
In procedures menu select IAP
In select airport menu, highlight the correct airport and confirm selection with the Select Airport Button
In the select procedure menu, highlight procedure. The procedure appears in cyan on the MFD. Confirm procedure with the Select Proc button

Source: SFB Pilot's Guide

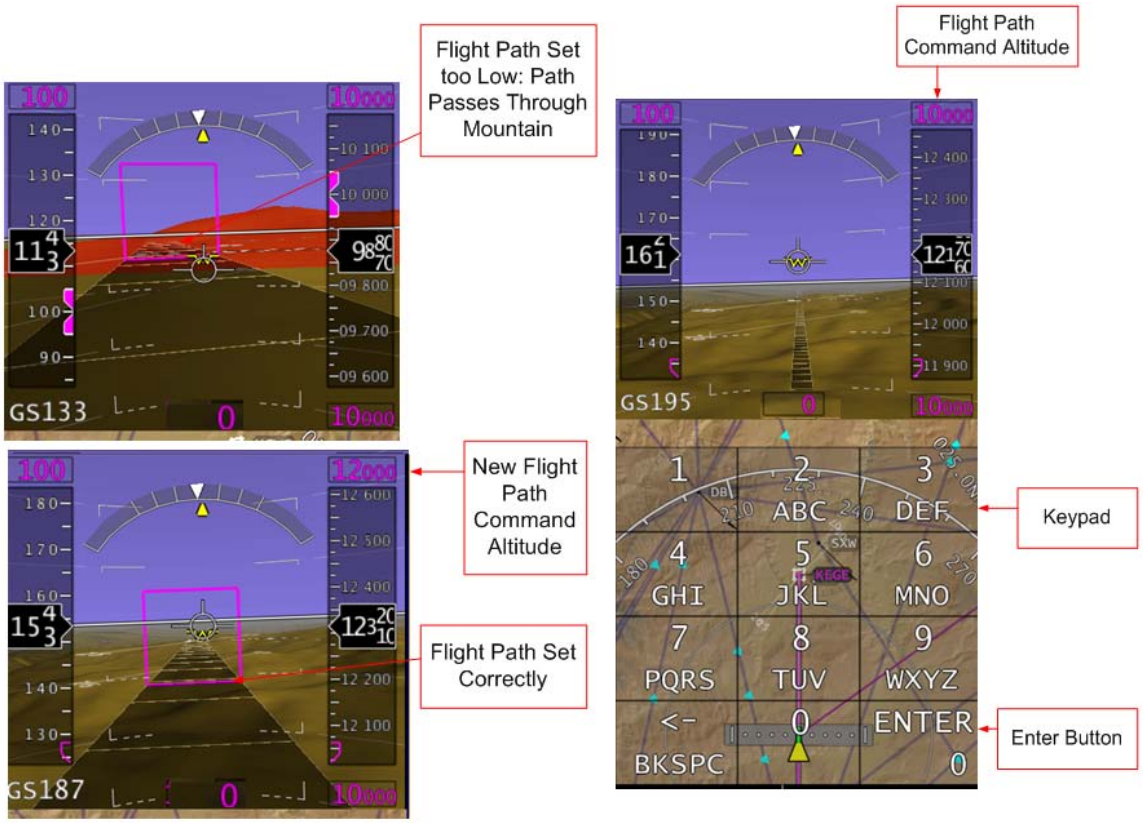


Figure 54 Setting the Flight Path Altitude

Note: To reset flight path altitude, select the flight path command altitude on the PFD On the keypad, enter the new path altitude and select enter

Source: SFB Pilot's Guide

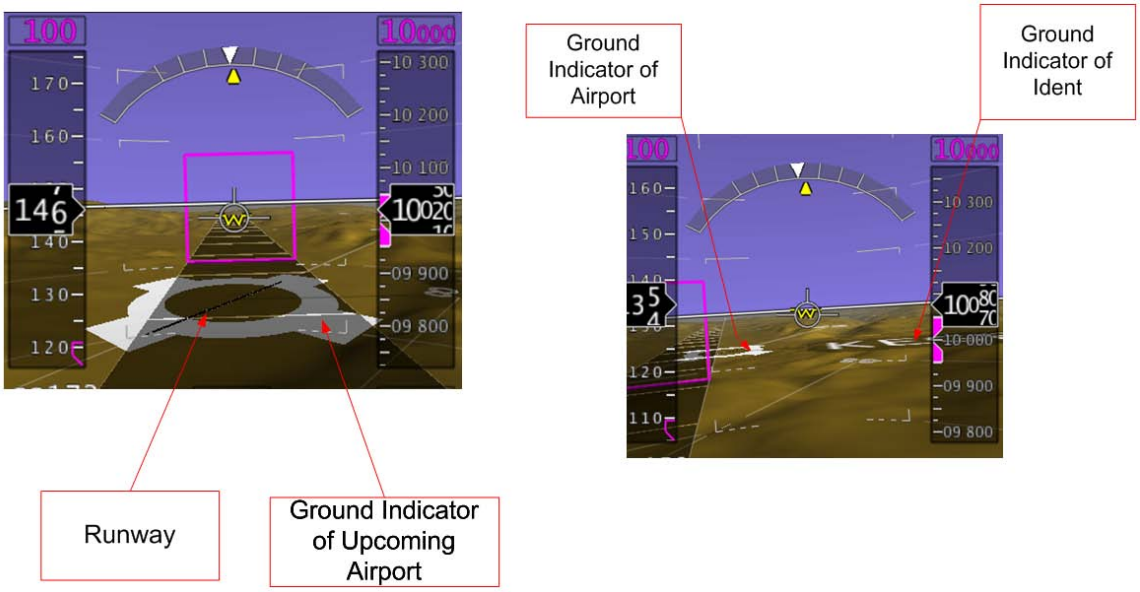


Figure 55 Ground Indicators

Source: SFB Pilot's Guide

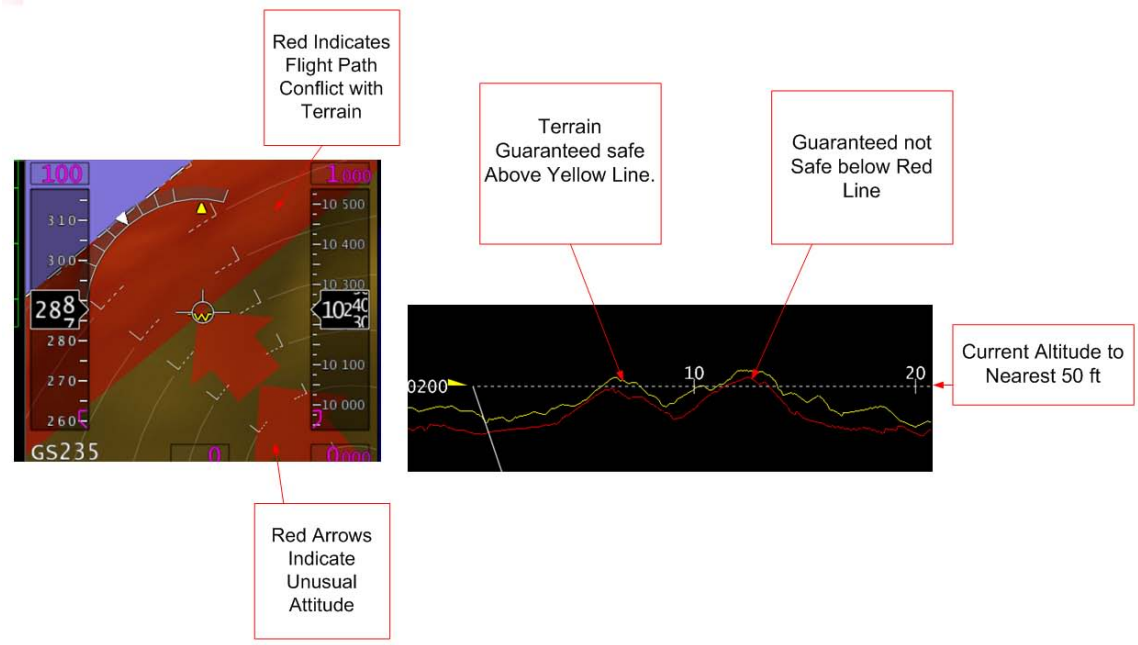


Figure 56 Terrain Awareness

Source: SFB Pilot's Guide

INTEGRATION

For the purposes of testing, it is necessary to be able to fly the Synthetic Flight Bag software in a reasonable aviation environment. To this end, integration was done in a variety of environments. The software was integrated with a plug-in to Microsoft Flight Simulator 2004 in order to provide reasonable aircraft state data for the purpose of debugging and testing during development. Also, several AHRS and GPS solutions were tested for their suitability for flight test, and the Dynon EFIS-D10A and uBlox RCB-LJ were selected as the sensors for use in the flight test.

The following parameters are used by the Synthetic Flight Bag software:

- ECEF x,y,z position of aircraft
- Latitude and longitude of aircraft
- GPS and barometric altitudes
- Indicated airspeed
- Ground speed
- Vertical speed
- True and Magnetic heading
- True and magnetic track
- Pitch and Roll
- Angle of attack, angle of Sideslip
- Flight path angle

Note that many of these parameters are redundant. For example, flight path angle can be computed from vertical speed and ground speed, or true heading can be computed from magnetic heading and latitude and longitude. These, however, are the set that are most useful. When driven from the simulator, all are filled with values driven from the simulator. When used on the flight test aircraft, however, many of these redundant values are derived mathematically from other values.

Flight Simulator Integration

A variety of aircraft state parameters are needed for effective use of the Synthetic Flight Bag software. Microsoft Flight Simulator 2004 was used as a flight model, and Peter Dowson's FSUIPC module was used to access internal parameters of the simulator.

The FSUIPC plug-in was installed in the Flight Simulator 2004 directory, and runs as an add-on as part of the simulator process. A library was built which connects via shared memory interface to the FSUIPC plug-in, allowing parameters to be queried. If a developer is running test builds on a laptop, the simulator with FSUIPC plug-in, and the Synthetic Flight Bag software may all be run on one computer. The communication library is built into the Synthetic Flight Bag software and communicates directly with the flight simulator.

Alternatively, a small external program that uses the same communication library is also used. In this case, the queries are sent to the flight simulator in the same way, but once received, the results of these queries are packaged into a UDP packet and sent via an Ethernet interface to the Synthetic Flight Bag hardware. This allows for testing of the software on the target hardware, using the touch-screen.

This has also been highly useful as a training tool, as a simulator was built using the ELITE Pro Panel II Flight Console, the Project Magenta General Aviation IFR Panel software, and a small frame. Two computers are stored in the back of the frame. One computer runs Microsoft Flight Simulator 2004 communicates with the Elite flight console, and provides an out-the-window view on a monitor attached to the back of th simulator frame. The second computer runs the IFR panel software, and its visual output is sent to a monitor mounted above the flight console. This monitor is hidden behind a plastic panel with accurately sized and reasonably placed holes cut through it, such that it forms a simulation of a standard instrument panel of a generic single-engine IFR-capable aircraft.

The gauges on the IFR panel software are sized and positioned to show in the holes in the panel, and provide a very realistic simulation. See Figure 57 on page 92. Additional panel space to the right of this display allows for a Synthetic Flight Bag display to be mounted to the panel. Connections are available with 12V DC power and Ethernet to connect the Synthetic Flight Bag computing box to the simulator as a whole.

Figure 57 Ground simulator

Note: At bottom in black is the Elite Pro Panel and Project Magenta IFR gauges behind plexiglass sheet. The round gauges seen are images on a LCD monitor, viewed through holes cut in the plexiglass cover sheet A Synthetic Flight Bag Display is mounted at right. At far right are two connectors providing power and aircraft state inputs to a Synthetic Flight Bag computer, just out of view to the right. Above the panel is the outside visuals monitor.

With this setup, ground training with the software can be accomplished, allowing subject pilots to be adequately familiarized with the system prior to their runs in the flight test aircraft.

Flight Test Aircraft Integration

The flight test aircraft selected was a Beech Bonanza. For the purposes of the flight test, the aircraft was changed to an experimental category registration, allowing for modifications to be made to the aircraft that would normally require an expensive and difficult field approval process. As an experimental category aircraft, it is possible to

make modifications to the aircraft using a much more streamlined process. In preparation for flight test, a number of modifications were made.

In its normally provided configuration, the Bonanza is equipped with only one control yoke, on an arm which may be pivoted to place it in front of the pilot or copilot as needed. In the interest of safety, it was deemed necessary for the experimental pilot and safety pilot to both have access to the controls, therefore the single yoke was removed, and replace with a dual yoke system, designed and approved for the Bonanza. This modification was approved through the standard field approval process. This

Figure 58 Flight test aircraft cockpit after modifications.

Note: The dual yoke install is visible, along with the newly added yoke at right. The installed supplemental instrument panel is visible, in beige, to the right of the radio stack and beside the door hinge. A dual-screen Synthetic Flight Bag setup is mounted in front of the pilot's yoke. View from the right wing looking down into the cockpit.

modification becomes a permanent change to the aircraft, and was done prior to the aircraft being placed in the experimental category. See Figure 58 on page 93.

Mounting tabs for a dual-display Synthetic Flight Bag system were added to the pilot's side of the main instrument panel, to allow a frame holding two displays to be mounted. This frame may contain two Synthetic Flight Bag displays. Another use of this frame is as a camera mounting point for flight test recording. These tabs were installed with the aircraft certified in the experimental category. See Figure 59 and Figure 60 on page 95 which show the instrument panel both before and after mounting of the dual

Figure 59 Pilot's instrument panel.

Note: Orange tabs are mounting hard-points for the SFB displays when installed as a pair. Throughout all photos, equipment installed under experimental-category approval is painted orange, with the exception of the equipment rack, which is approved through the standard field approval process. View from the pilot seat.

Figure 60 Pilot's instrument panel with dual-SFB displays installed

Note: When installed, the displays cover the six primary instruments, along with the clock, the radio altimeter, and the alternator failure and door-ajar annunciator lights. Also partially covered is the autopilot control panel. View from the pilot seat

display frame. The frame is removable in flight, should it be necessary for flight safety.

The middle seats of the aircraft were removed from their mounting rails, and an equipment rack was installed in their place. This equipment rack was built to match designs of equipment racks used at the FAA Technical Center in Atlantic City, NJ. Finite element analysis was performed to verify the design of the rack, and this supporting analysis was used to get a field approval for the installation of the rack. This modification was done prior to the aircraft being placed in the experimental category, and as the rack does not modify the aircraft itself, this rack can be used even if the aircraft is

eventually returned to a standard-category aircraft. See Figure 61 and Figure 62 on pages 96 and 97.

A small glove-box on the right side of the instrument panel was removed, and a supplemental instrument panel installed in its place. Within this instrument panel, a second airspeed indicator and altitude indicator were added. This installation was also done with certified instruments and approved through the standard field approval process. This modification becomes a permanent change to the aircraft, and was done prior to the aircraft being placed in the experimental category.

Figure 61 Equipment rack on the middle-row seat rails in the flight test aircraft.

Note: The bottom shelf contains 2 Synthetic Flight Bag computers, along with the aircraft I/O computer. The box on the top shelf contains a Novatel GPS unit. View from outside the aft doors on the right side of the aircraft, looking forward into the cabin.

Figure 62 Equipment rack as installed on the middle-row seat rails.

Note: Blue and grey cords are Ethernet. A small Ethernet switch is attached to the bottom side of the top shelf. At the left edge of the image is the aircraft connection panel. View from the copilot's seat, looking down and back towards the cabin and tail

A Dynon EFIS-D10A system was installed in the aircraft. As the Dynon system is designed for experimental and homebuilt aircraft, as it is directly connected to the aircraft power systems, and as it displays attitude information to the pilot, there is a danger that it may mislead the pilot if it fails. For this reason, it would be difficult or impossible to receive a field approval to install this system in a standard-category aircraft. For this reason, this and all subsequent modifications were done with the aircraft certified in the experimental category. The display head of the Dynon system was installed in the supplemental instrument panel on the right side of the aircraft, as seen in Figure 63.

Figure 63 Supplemental instrument panel with Dynon EFIS-D10A installed.

Note: The Dynon is the large square display at center with grey buttons. Below are standby altitude and airspeed indicators. To the right is a mechanical g-load meter. View from the copilot's seat.

An inspection cover plate on the right wing was removed, and modified as a support for the pitot probe. Static pressure was taken from inside the aircraft cabin, as the Bonanza is not a pressurized aircraft. It was necessary to install a separate pitot and static system in the flight test aircraft, as the existing pitot and static system has been tested and approved for IFR flight. Attaching to the existing pitot and static system may be possible, but the modifications made in doing so would have invalidated the existing IFR approval. Standard aircraft pneumatic tubing was used to plumb the pitot probe to the Dynon. Figure 64 shows the pitot probe installed on the flight test aircraft.

Figure 64 Dynon pitot and angle of attack probe fitted below the right wing.

Note: Orange plate was an old inspection port cover, modified as a support for the probe. The white panel at far right, hanging down from the wing is the right landing gear door. View from in front of the right wing, looking aft and to the left of the aircraft, toward the fuselage.

As the Dynon includes a magnetic compass, it is important that the compass be well isolated from any electrical fields in the panel or engine. For this reason, it is common to mount the compass itself far from any possible interference. The wingtip and tail are common mounting locations. A small bracket was fabricated, and a remote magnetometer was installed in the right wingtip, as can be seen in Figure 65. Signal cable was run down the span of the wing, and connected to the display head in the panel. Additionally, an Outside Air Temperature (OAT) probe was mounted in the wingtip. This probe is used to correct the barometric readings for the effect of temperature.

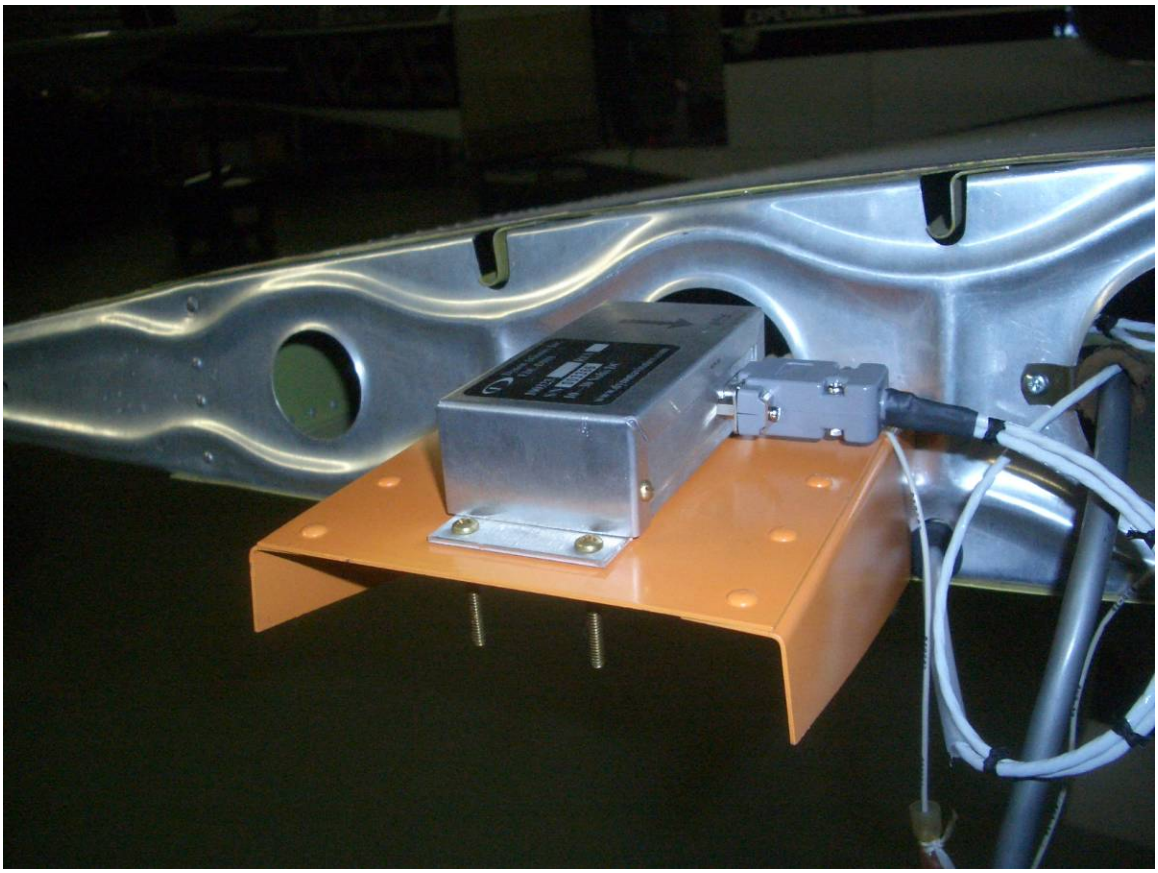


Figure 65 Dynon remote magnetometer as installed in the right wingtip.

Note: The orange bracket was fabricated as a mount for the magnetometer. Cabling, white, runs down the wing to connect to the display head. The OAT probe is on the lower surface of the wing, hidden behind the bracket. View from the tip of the right wing, looking left toward the fuselage. Wingtip cover removed.

String potentiometers were added to sense the position of the ailerons, elevator, and rudder control cables. These potentiometers have a cable reel attached, from which a thin steel cable unwinds. Spring tension winds the cable back into the reel when it is slack and keeps the cable under tension. The reel itself is connected to a potentiometer, which provides a varying resistance as a function of cable movement. The far end of the cable was attached using mounting blocks to the aircraft control cables, and wiring for the potentiometers was installed. These potentiometers are used to record the pilot's control inputs during flight test. See Figure 66.

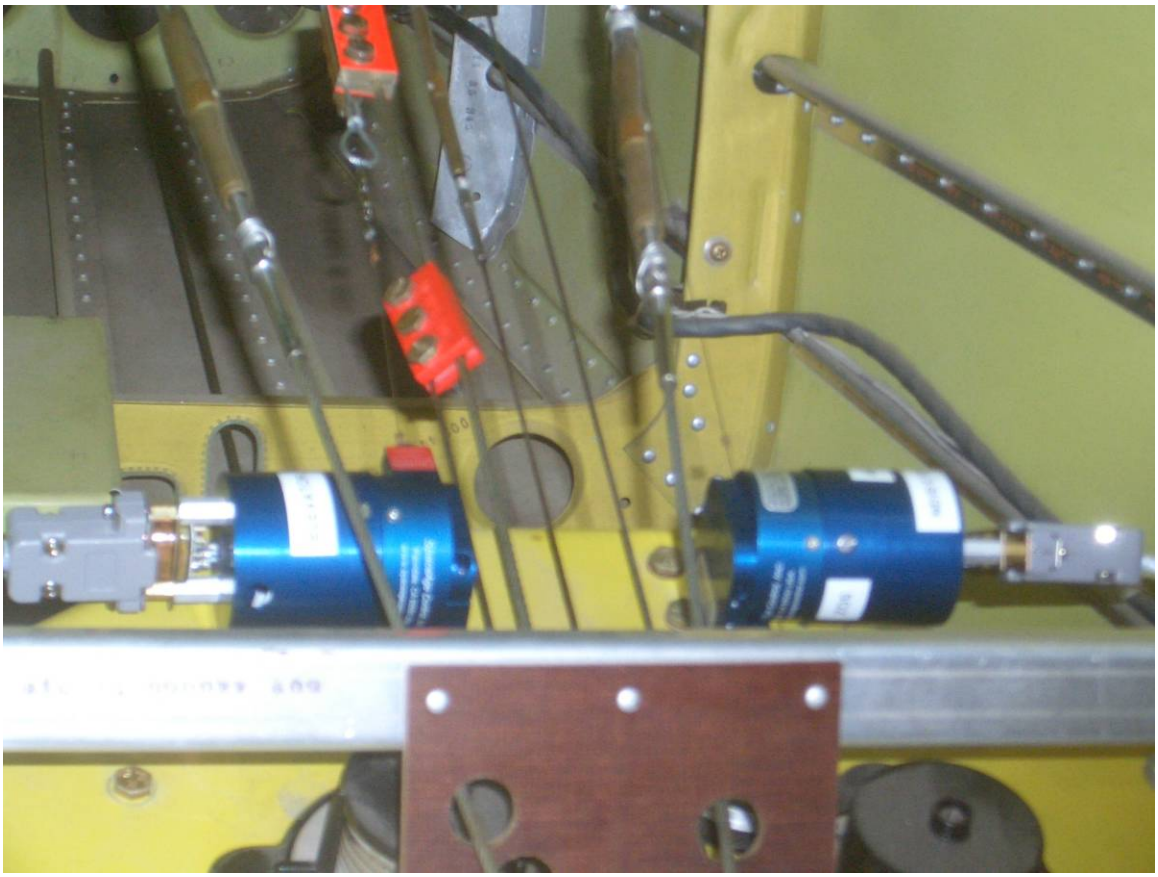


Figure 66 String potentiometers as installed.

Note: Potentiometers are the blue cylinders. Viewing the left potentiometer, the cable drum is on the right. The red square label marks the exit point of the cable, and the cable runs aft to the eye loop near the top of the image. View from the rear of the cabin, looking aft into the tail, with the rear bulkhead removed.

An aircraft connection panel was installed at the front edge of the middle row seat rails as can be seen in Figure 67. This panel contains a number of connectors, which provide connections for aircraft power, taken from the main aircraft bus through a 30 amp breaker. A second connector attaches to an extra GPS antenna installed in the aircraft, and is used to connect to a Novatel GPS receiver installed in the equipment rack. A third connector provides a connection to the aircraft's intercom system, allowing for the recording of the cockpit audio during the flight test. Finally, a fourth connector provides signals from the aircraft. This connector includes the Dynon communication

Figure 67 The aircraft connection panel.

Note: Left to right on the bottom row are the power feed, the aircraft audio connection, and the aircraft data connection. Top row connector at right is the GPS antenna feed. The other connectors are reserved for future use and unused. View from behind the copilot's seat, looking down and aft.

cables for the purpose of receiving aircraft state information, connections to the control measurement potentiometers, and a connection to the aircraft's existing radio altimeter.

An additional set of signal cabling was added, connecting to the safety pilot's yoke. This cabling was installed in anticipation of the need for event marker switches or other controls. In the end, this cabling was not used in the flight test.

Once all these connections were in place, an additional dedicated I/O computer was used to integrate all these various sources. This computer contains interface boards to convert the analog voltages from the potentiometers to sampled digital signals, serial interfaces to connect to the Dynon EFIS-D10A and the Novatel GPS receivers, and combines the data collected through various means into a single data stream. This data is logged to disk for the purpose of future analysis, and is broadcast to the Synthetic Flight Bag computers in the same way that the data is sent from the ground simulator. This allows Synthetic Flight Bag computer boxes to be used interchangeably on the simulator and in the flight test aircraft itself. The I/O computer can be seen installed on the leftmost rack of the bottom shelf in Figure 62 on page 97. The I/O computer has a red power button on its face and can be seen connected with cables to the aircraft connection panel. The two computers to the right of the I/O computer are Synthetic Flight Bag computers and receive aircraft data via Ethernet interfaces.

DISCUSSION AND CONCLUSIONS

A flight test was designed and executed to test the usability of the Synthetic Flight Bag system. The OPL Computerized Airborne Research Platform (CARP), an instrumented experimental A-36 Beechcraft Bonanza was used for the flight test. The CARP was fitted with the necessary instrumentation to quantify physiological and flight state data for the purpose of evaluating flight technical performance resulting from using the SFB system.

Flight Test Scenario

The flight test was performed using an evaluation pilot (EP) at the left front crew station, a safety pilot (SP) at the front right crew station, and a flight test engineer (FTE) at the aft crew station. The evaluation pilot crew station had a removable vision restriction device (VRD) installed, so that the EP could not see out the front windshield. The VRD did not interfere with the SP's duties to see and avoid other traffic in the area. Flight following service was provided by the Cedar Rapids and Quad Cities Tracon during all phases of the flight test.

The SP was the commander of the ship during all sorties and he was in charge of overall flight safety. The SP taxied the aircraft and performed the take-offs and landings. The FTE was in charge of the experimental payload, which included getting the aircraft systems ready for data collection, perform the calibration of the eye tracker, and collect the actual data during the sortie. After each flight, he also extracted the data off the data collection computers.

The scenario involved an investigation of controlled flight into terrain behavior using proxied terrain. This means that the Synthetic Flight Bag system terrain database was modified to contain high mountains South of Iowa City. The database was modified by moving terrain from Bakersfield, CA, into the Iowa City and Riverside, Iowa area. When flying in the Iowa City area with the modified terrain database, the SFB system

showed a chain of tall mountains South of Iowa City with the long dimension of the chain extending in the East-West direction. North of Iowa City, the modified terrain database showed terrain that was essentially flat. The experimental flight scenario involved a cross country flight from Iowa City (KIOW) via direct to the Cedar Rapids VOR (CID) and then via V67 to the Waterloo VOR (ALO). As this segment lead over relatively flat and low terrain, an enroute altitude of 4,500 ft (MSL) was programmed.

A modified navigation database was also generated in conjunction with the modified terrain database. The Iowa City Municipal Airport (KIOW) was renamed to Home City Municipal Airport (KHCM). Cedar Rapids was renamed to Western Agro (WAG), and Waterloo was renamed to Napoleon (NAP). Most of the surrounding airports and navaids were changed to fictitious names as well, to give the pilot a sense that he was flying in another area. Proxied paper charts were painstakingly generated based on the changes in terrain and navaid names, so that both the SFB system indications as well as the paper charts were in agreement. Since the modified area consisted of elements from Iowa and California, we called this fictitious area “Caliowa”. The reason for creating this proxied scenario was to allow the study of controlled flight into fictitious mountains with no risk of actual terrain proximity. Also, all navaids such as VORs were fully functional using the standard onboard navigation systems, allowing the runs to be performed identically using the standard aircraft instruments as a baseline configuration. The EPs were thoroughly briefed so that they knew that we were going to operate in a fictitious place called “Caliowa” and that no reference to any feature in the actual area of Iowa was to be made. This meant that the EPs were to use only information from the modified SFB system databases and from the modified paper charts to plan and execute the flight in the fictitious “Caliowa” region. Figure 68 on page 106 illustrates the modified VFR chart for the “Caliowa” flight area, and Figure 69 on page 107 illustrates the modified IFR chart, which was used in the flight test.

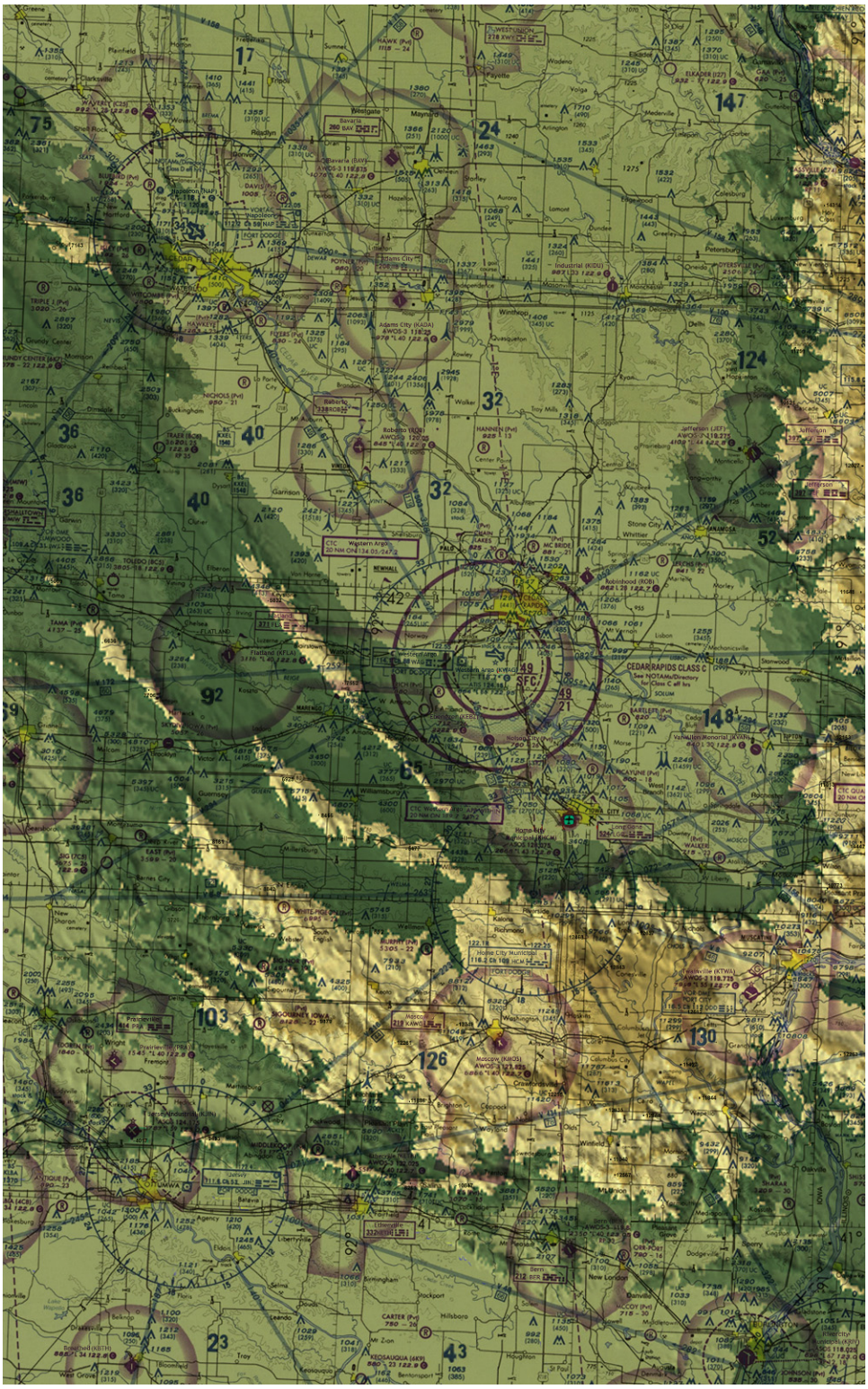


Figure 68 VFR Sectional Chart of "Caliowa"

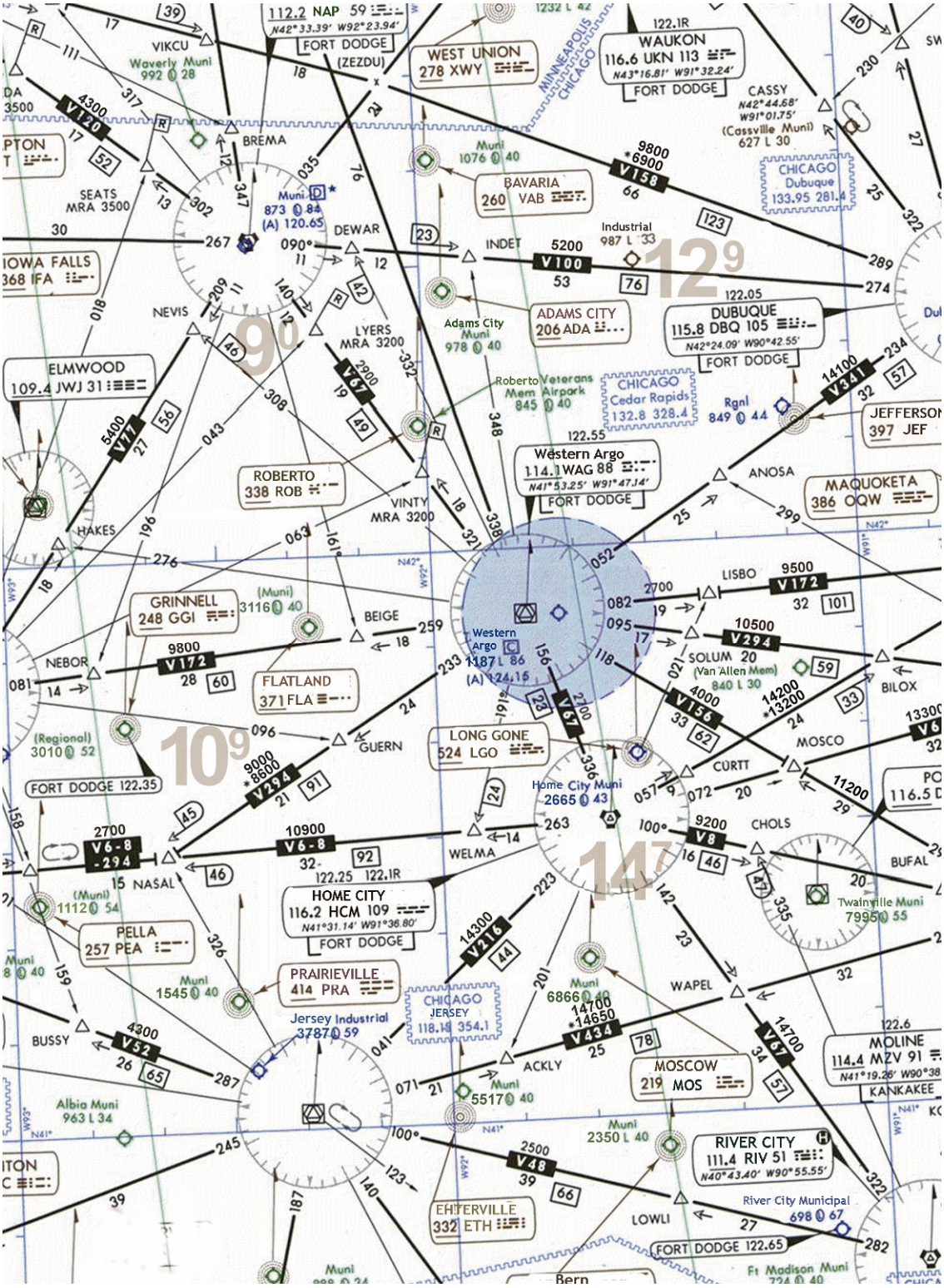


Figure 69 IFR Low Altitude Chart of “Caliowa”

There were two equally sized strata of EPs, Baseline and Synthetic Flight Bag. EPs in the baseline stratum had only a paper chart and the Baseline Round Dials (BRD) instruments. EPs in the SFB stratum had access to BRDs and to the SFB device that showed a graphical representation of the terrain. In the Baseline stratum, terrain avoidance was possible only by reference to Minimum Enroute Altitudes (MEAs) on the modified "Caliowa" navigation chart. In the SFB stratum, EPs were able to see the rising terrain South of KHCM on the screen of the SFB system.

This concept was also briefed to the FAA in a visit to the Cedar Rapids Tracon, as they needed to provide the flight with traffic separation. A sequence of phases that described out intent was developed in collaboration with the FAA. For example, phase I consisted of aircraft familiarization West of the KHCM airport, Phase II consisted of intercepting a course from KHCM to WAG and an implied climb to 4,500 ft (MSL). This way, it was possible to indicate the intent to intercept the course from KHCM to WAG with a simultaneous climb to 4,500 ft (MSL) simply by calling ATC with the phrase "N23540 beginning Phase II". ATC acknowledged this request simply with "Roger". Using this system of phase numbers that identify portions of a known experiment script, it was possible to minimize communication traffic with ATC. During the experiments, ATC also kept traffic out of the volume of airspace needed for free vertical navigation by the EP.

As the flight progressed a few miles past the WAG VOR, the EP was given a surprise deviation to the South to the Burlington VOR. We renamed the Burlington VOR to River City (RIV). This deviation was coded as Phase III and ATC would expect the flight to make a left or right turn to the South and potentially start a climb to avoid the fictitious range of tall mountains South of KHCM. Again, all this could be communicated to ATC simply by a radio transmission that Phase III was being initiated.

One important safeguard of the airspace system was removed for the purpose of the flight test. Normally, pilots communicate their intended flight plans and altitudes to

ATC, and are cleared for those plans. It is ATC's responsibility to ensure the pilot has chosen an appropriate and safe flight altitude for the flight plan they have chosen. In the event that the flight plan changes, ATC will check the new flight plan for this same reason. Nonetheless, through human error, this step is sometimes forgotten or done incorrectly. For the purpose of the flight test, the pilot was not informed that their flight altitude was inappropriate, simulating an error by ATC. Thus the test was to see if the experimental pilots were able to safely recognize this error and the need for a change to their flight altitude to clear the mountains to the south of Home City, or if they would continue at their original altitude and crash in the mountains.

Following the diversion, the SP reminded the EPs to navigate directly to the RIV VOR using the onboard VHF radio system, and in the case of pilots in the SFB stratum, by additional reference to the SFB system. In addition, EPs were reminded that they were in "Caliowa", that all navigation information was to be derived from the "Caliowa" chart or from the SFB system, and that any and all vertical navigation was automatically approved as deemed necessary by the EP without any clearance restriction or need for communication with the SP or ATC. No indication that a climb was necessary was given. After this verbal reminder, the SP was quiet and watched the progress of the flight with regard to avoidance of the "Caliowa" mountain range. Phase III ended either when the FTE confirmed from his navigation display that the EP had crashed into the fictitious mountains or when the EP had detected the ensuing terrain conflict and offered or implemented a strategy for resolution.

Following Phase III, the proxied "Caliowa" database was cleared and the normal Iowa database was loaded into the SFB system. ATC was informed that Phase III was complete and standard navigation charts and approach plates were made available to the EP for the ensuing instrument approach phases.

Each EP flew two ILS 24 approaches at Muscatine (KMUT) and two VOR approaches. One ILS approach was flown on BRD and one was flown with additional

access to the SFB system. The order of the BRD or SFB usage was blocked between EPs to avoid any sequence effects. The first VOR approach was VOR-6 at KMUT and the second one was the VOR-A at KIOW. One of the VOR approaches was flown on BRD only and the other one was flown on BRD with additional reference to the SFB system. The assignment of the SFB system to the type of VOR approach was blocked between pilots. Two holding patterns were flown during each sortie, one at the Port City VOR (DDD) at the KMUT airport and at the IOW VOR. Assignment of BRD and the additional SFB system to each holding task was blocked between EPs.

We selected ILS and VOR approaches to contrast flight technical performance for precision and non-precision approaches. The hypothesis was that the SFB system would be especially effective in tightening flight technical performance during non-precision approaches. An additional hypothesis was that for pilots who were new to this type of technology, the SFB system would require a fair amount of eye fixation time during high workload operational use. During the approaches the SP provided the EP with frequency tuning and SFB programming assistance. The EP simply had to ask for an approach to be loaded into the SFB system or a frequency to be dialed into the radio and the SP would perform this service.

The full results of the flight test are published in a NASA final report, (Schnell and Wenger, 2007). Several of the more important points, however, are presented here.

Simulated CFIT Accident Results

First, the results indicate that having the Synthetic Flight Bag system available to the IFR pilots significantly reduced the incidence of CFIT accidents, as only one of six pilots with traditional paper charts noticed the ATC error and survived, whereas five of six pilots using the Synthetic Flight Bag system saw and avoided the terrain they were approaching. This result can be seen in Table 3 on page 111. Analysis of the performance of pilot 12 indicates that at the time of the simulated CFIT accident, the pilot

Table 3 Controlled flight into terrain flight test results

Pilot	SFB Usage Time		Terrain Clearance
1	No SFB (Control)		None (CFIT)
3	No SFB (Control)		None (CFIT)
5	No SFB (Control)		15.93 nm from terrain
7	No SFB (Control)		None (CFIT)
9	No SFB (Control)		None (CFIT)
11	No SFB (Control)		None (CFIT)
2	654.27 sec	70.8% of run	25.56 nm from terrain
4	712.68 sec	50.8% of run	6.74 nm from terrain
6	236.00 sec	16.0% of run	3.09 nm from terrain
8	173.03 sec	12.0% of run	7.68 nm from terrain
10	1057.95 sec	55.8% of run	27.56 nm from terrain
12	267.30 sec	10.4% of run	None (CFIT)

Note: Of the six pilots control group who used the traditional aircraft instruments, all but one experienced a controlled flight into terrain when rerouted.

Of the six pilots using the SFB, all but one saw the need to climb to avoid terrain.
Pilots 2 and 10 started climbing immediately after turning to the new path, presumably via the look-ahead capabilities of the VPD.
Pilots 4, 6, and 8 began climbing 3-8 miles from the rising terrain, presumably having been alerted by terrain warning colors on the PFD.
Pilot 12 experienced a controlled flight into terrain when rerouted.

was under high workload as part of the navigation toward River City, and did not observe or understand the terrain warnings being presented by the Synthetic Flight Bag display. This indicates that while the system may help in improving flight safety, it is not, in its current form, a “silver bullet” solution to the problem of controlled flight into terrain.

Flight Technical Error Results

One measure of the accuracy with which a pilot flies an approach is cross track error, seen as a side-to-side error in the tracking of the centerline of the route. Results indicate that the Synthetic Flight Bag system considerably reduces cross track error on both precision and non-precision approaches. Vertical track errors, or height errors, were of similar magnitude for the BRD and for the BRD+SFB configurations, respectively. These results may be seen in Figure 70 on page 112 and Figure 71 on page 113.

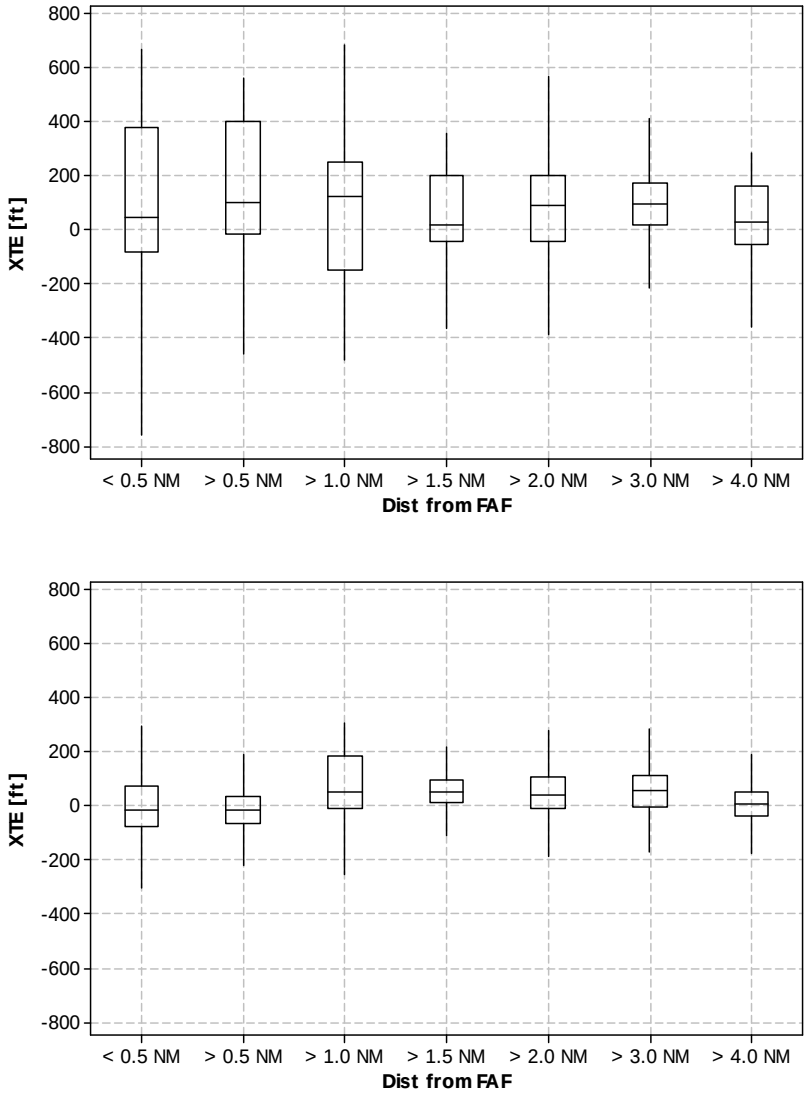


Figure 70 XTE versus distance from FAF, ILS 24 at Muscatine, IA

Note: The figure at top illustrates cross track error using the baseline round dial instruments, on the ILS 24 approach into Muscatine, IA. The bins, represent increasing distance from the Final Approach Fix, and decreasing distance to the runway threshold.

The figure at bottom illustrates the ILS 24 task using the baseline round dial instruments with the addition of the Synthetic Flight Bag system. With the Synthetic Flight Bag system added, cross track errors have been significantly reduced compared to the baseline, indicating that pilots fly more accurately using the Synthetic Flight Bag.

Also noteworthy is the fact because the ILS measures angular error relative to the runway, the guidance of the ILS becomes more sensitive as the runway is approached, as can be seen by the lower errors as a function of time. By contrast, the SFB guidance is linear, with consistently sensitive guidance. It was observed that some pilots appeared to perform better with this constant sensitivity, compared to the rising sensitivity of the ILS.

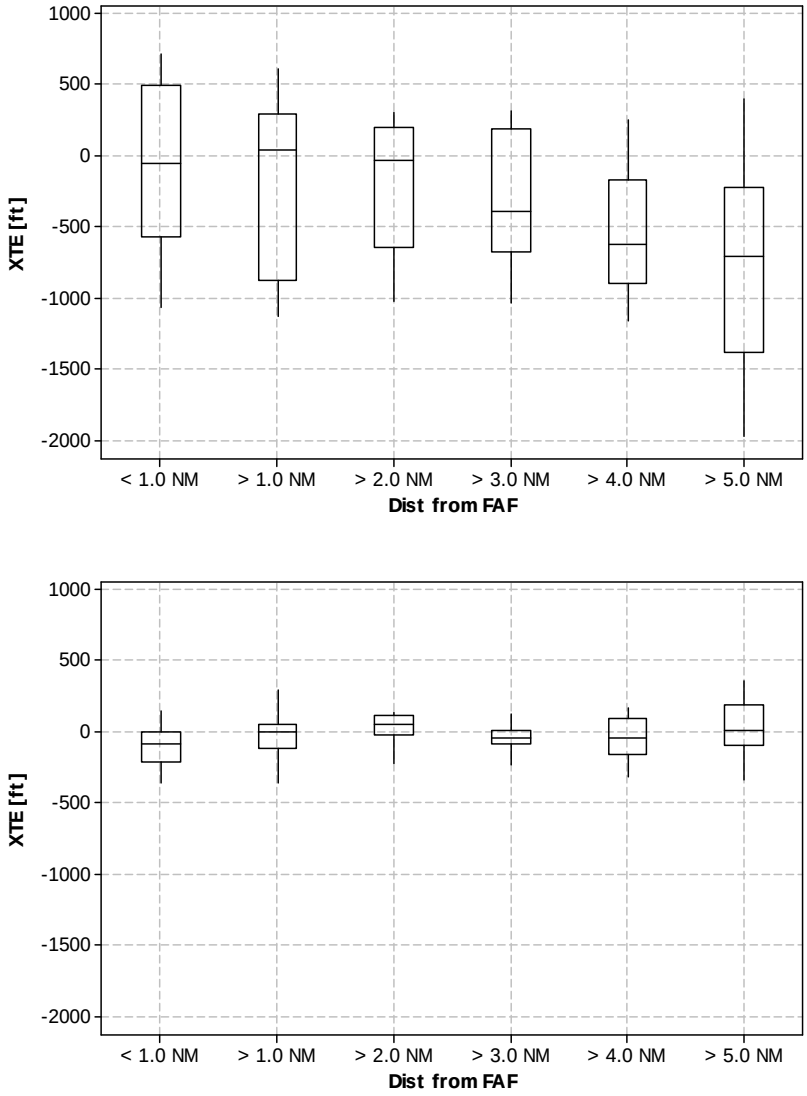


Figure 71 XTE versus distance from FAF, VOR-A at Iowa City, IA

Note: The figure at top indicates cross track error using the baseline round dial instruments on the VOR-A approach into Iowa City, IA. Cross track error on this non-precision approach is much higher than on the precision approach of ILS 24 in Muscatine.

The figure at bottom indicates the VOR-A task using the baseline round dial instruments with the addition of the Synthetic Flight Bag system. Flight technical errors are dramatically lower than the baseline, and remain at comparable levels to that of the precision approach, even when using the far less sensitive guidance provided by the VOR approach.

Future Work

Anecdotally it seems that devices such as the SFB system have the potential to provide significant safety benefits, especially with regard to terrain avoidance. However, it should be noted that efficient use of complex flight planning functions, vertical navigation functions, and instrument approaches require considerable training and familiarization. Such features may in fact increase workload during early stages of use, until a fair amount of operational experience has been amassed. Ideally, pilots would gradually phase in use of such systems during levels of low workload. Proper familiarization and training will be an important need in any future version of the system.

Several features were identified as important for a future version of the Synthetic Flight Bag system. The first is aural alerting. A connection to the aircraft intercom system would allow the Synthetic Flight Bag to announce audible alerts of terrain conflict to the pilot, which would likely give sufficient warning to ensure that a controlled flight into terrain accident would be avoidable. The current system as implemented does not necessarily do enough to draw the attention of an otherwise distracted pilot.

Another potential feature would be to extend the red terrain alerting to the multifunction display. It was initially conceived that terrain avoidance was a tactical task suited for the PFD, rather than a strategic task suited for the MFD, and that any strategic terrain avoidance would be done with the aid of the VPD. The flight test demonstrates that with the level of training and familiarity with the system, pilots in the flight test were not able to reliably predict terrain conflict using the VPD. Adding a similar function to the MFD may improve the chances of a terrain conflict being noticed.

Finally, additional automated checking on the pathway to ensure that the pilot has selected a cruise altitude that is appropriate for the flight plan, would also provide an appropriate increase in safety. In this case, the route entered would be checked against published airway altitudes, and warnings would be issued when the selected altitude is less than that required as part of the published Minimum Enroute Altitudes.

REFERENCES

Keller M., Schnell T., Lemos K., Glaab L., Parrish R. (2003) *Pilot Performance as a Function of Display Resolution and Field of View in Simulated Flight Using Synthetic Vision Systems*. Proceedings of 22nd Digital Avionics Systems Conference, Dawn of the 2nd Century / Racing to Transform the Legacy, The Crowne Plaza, Indianapolis, Indiana, 12-16 October

Lemos K., Schnell T., Etherington T., Gordon D. (2002) *Bye-Bye Steam Gages, Welcome Glass'; A Review of New Display Technology for General Aviation Aircraft*. In proceedings of 21st Digital Avionics Systems Conference, Air Traffic Management for Commercial and Military Systems, Hyatt Regency, Irvine, California, 27-31 October 2002

Lemos K., Schnell T. (2003) *Synthetic Vision Systems: Human Performance Assessment of The Influence of Terrain Density And Texture*. Proceedings of 22nd Digital Avionics Systems Conference, Dawn of the 2nd Century / Racing to Transform the Legacy, The Crowne Plaza, Indianapolis, Indiana, 12-16 October 2003

Miller, P.A. (1998) *Recursive Make Considered Harmful*, AUUGN Journal of AUUG Inc., 19(1), pp. 14-25

Schnell, T., Lemos, K., Keller, M., Yang, S. (2003) *Synthetic Vision Systems, Optimum Display Characteristics.* Final Report, NASA Langley Research Center, Aviation Safety Program, Hampton, VA

Schnell, T., Wenger J., (2007) *Advanced Media, Portable Media*, Final project report, Contract NNL04AA22G, NASA Langley Research Center, Hampton, VA, 2007

Stallman, R. M., McGrath, R., Smith, P. D. (2004) *GNU Make Manual*. Boston, MA: Free Software Foundation