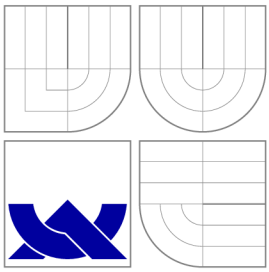
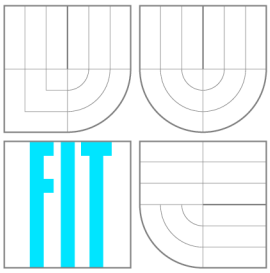


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AEROWORKS: GRAFICKÉ ROZHRANÍ SYSTÉMU AUTOPILOTA

AEROWORKS: GRAPHIC USER INTERFACE FOR AUTOPILOT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — VIKTORIYA DURYAGINA
AUTHOR

VEDOUCÍ PRÁCE — Ing. PETER CHUDÝ, Ph.D. MBA
SUPERVISOR

BRNO 2012

Abstrakt

Tato bakalářská práce se zaměřuje na návrh grafického uživatelského rozhraní pro obsluhu autopilota. První část práce obsahuje krátký přehled teorie automatických systémů pro řízení letu. Dále jsou analyzovány požadavky na rozhraní a prozkoumány problémy komunikace mezi pilotem a systémem. Následně je popsán proces návrhu a vývoje nového grafického rozhraní systému. Výsledné řešení je implementováno do simulátoru v laboratoři SimStar.

Abstract

The objective of the bachelor thesis is to design a graphical user interface for autopilot control system. The first chapter is dedicated to automatic flight control system theory overview. The next part discusses the user interface requirements and human-computer interaction problems. Subsequently, the design process and the development of new user interface are described. The final solution is implemented in the SimStar simulator.

Klíčová slova

autopilot, automatické systémy řízení letu, grafické rozhraní, CANaerospace, AW-COM, laboratoř SimStar, AeroWorks

Keywords

autopilot, automatic flight control system, graphical user interface, CANaerospace, AW-COM, SimStar laboratory, AeroWorks

Citace

Viktoriya Duryagina: Aeroworks: Grafické rozhraní systému autopilota, bakalářská práce, Brno, FIT VUT v Brně, 2012

Aeroworks: Grafické rozhraní systému autopilota

Prohlášení

Prohlašuji, že tato bakalářská práce je mým původním autorským dílem, které jsem vypracovala samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používala nebo z nich čerpala, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

. . . . . .

Viktoriya Duryagina

14. května 2012

Poděkování

Na tomto místě bych chtěla poděkovat Ing. Peteru Chudému, Ph.D. MBA za odborné vedení této práce a za cenné rady a náměty. Dále bych chtěla poděkovat týmu laboratoře AeroWorks za pomoc a čas, který mi věnovali při zpracování této bakalářské práce.

Obsah

Kapitola 1

Úvod

V dnešní době jsou moderní letadla vybavena velkým množstvím leteckých přístrojů a automatů, které zajišťují nepřetržitou kontrolu režimu letu a řeší složité úkoly automatického řízení a navigace. Význam těchto nástrojů se každým rokem zvyšuje. Spolu s narůstajícím významem roste i celková složitost, rozsah a náročnost návrhu a vývoje těchto systémů. Systémy do sebe sdružují více informací, pilotům poskytují další nové funkce a nástroje pro ulehčení práce. Celé části informačních systémů se pak spojují do logických celků, do kterých přibývají nové moduly s novou funkcionalitou. I zkušený pilot musí před první letem s jiným letadlem absolvovat dlouhá školení a seznámit se s obsáhlými manuály pro ovládání a chování letadla.

Pro tyto důvody směřuje pozornost vývojářů zejména k vytváření jednoduchých, přehledných a intuitivních ovládacích rozhraní systémů. Při vývoji leteckého informačného systému je potřeba klást velký důraz na kvalitu návrhu grafického rozhraní tak, aby systémy pilotům poskytovali všechny potřebné informace jednotným způsobem a v přehledné podobě.

A právě návrhem a implementací grafického uživatelského rozhraní systému autopilota se zabývá tato bakalářská práce. V první kapitole práce je uveden krátký přehled historie a teorie automatických systémů řízení letu. Uvedeny jsou základní fyzikální principy, na kterých je založeno fungování autopilotů. Pro lepší pochopení problematiky pilotních systémů jsou také popsány základy pohybu letadla v prostoru a vysvětleny pojmy s tímto související. Další kapitola se věnuje obecným zásadám a problémům návrhu uživatelských rozhraní pro letecké aplikace. Popsány jsou také specifikace a požadavky na nově vznikající systém.

Hlavní část práce je věnována popisu návrhu a vývoje aplikace, zejména designu interakce – procesu komunikace člověka se systémem. Sledován je vývoj grafického uživatelského rozhraní, jednotlivých návrhových verzí systémů a jejich přínos pro celkový výsledek. Výsledek návrhu je následně implementován a nasazen do prostředí leteckého simulátoru v laboratoři SimStar. V poslední kapitole se pak nachází shrnutí výsledků práce a testování.

Kapitola 2

Teorie autopilotů

Spolu s prudkým rozvojem letadel na začátku 20. století se začínají objevovat a používat první systémy automatické kontroly a řízení letu. Potřeba automatizace řízení letu letadla byla vynucena nedostatečnou stabilitou a ovladatelností letadla – letadla bez automatických pilotů a systémů pro stabilizaci letu byla ve vzduchu málo stabilní a bylo nutné neustále upravovat správnou pozici letounu, tak aby bylo možné zachovat bezpečný let. Let na letadlech bez automatického řízení vyžadoval vysokou technickou úroveň pilotování, velkou koncentraci vnímání a soustředění pilota.

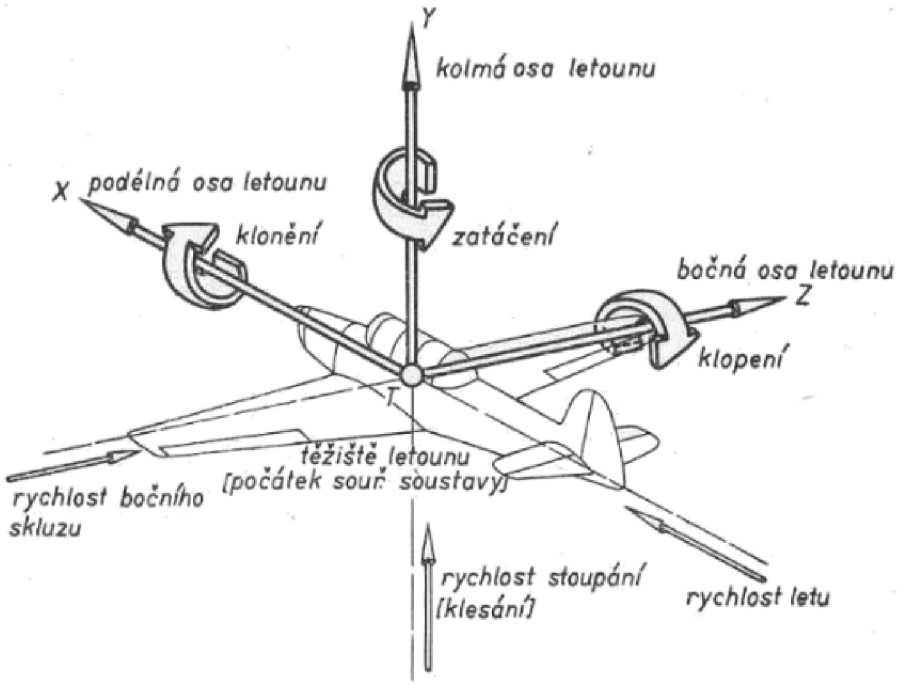
Moderní konstruování letadel se vyznačuje širokým používáním systémů automatického letu. Uplatnění automatických nástrojů je diktováno vylepšením leteckých charakteristik letadla, jeho stability a ovladatelnosti. V daný moment se systémy automatického letu jeví jako prostředek pomáhající pilotu ovládat letadlo.

Široká automatizace procesu ovládání letadla nevylučuje pilota z řízení, ale nechává pro něj funkce ovládání různých režimů autopilota, nastavení autopilota a také kontrolu procesu pilotování letadla.

Z důvodu nepřetržitého technického pokroku a zlepšování technických charakteristik letadel, které umožňovaly mnohem delší dobu pobytu letadla ve vzduchu, docházelo ke stále většímu zatížení pilota. Proto bylo nutné podniknout kroky k ulehčení práce pilota a posádky od unavujících a jednotvárných úkonů nutných pro stabilizaci a řízení letadla. A právě tyto důvody se staly hlavním impulsem pro vznik a rychlý rozvoj systémů automatického řízení letadla tzv. autopilotů.

2.1 Pohyb v prostoru

Letadlo koná v prostoru velmi složitý pohyb, který můžeme popsat v pravoúhlé souřadnicové soustavě jako posuvný pohyb se složkami posuvu ve směru všech tří os a jako otáčivý pohyb kolem těchto os [1]. Pro popis pohybu letadla v prostoru se používá letadlová souřadnicová soustava, která je pevně spojená s letadlem a jejíž počátek leží v těžišti letadla (obrázek 2.1).
Z hlediska pohybu letadla lze rozlišit tři základní pohyby kolem jeho os.



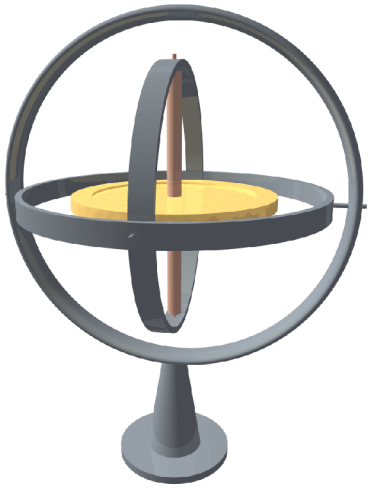
Obrázek 2.1: Letadlová souřadnicová soustava [2]

- **Klopení (pitch)** – pohyb letadla kolem boční osy, vedoucí přes letadlo od jednoho křídla ke druhému. Udržuje podélnou stabilitu letu – naklopení nosu letadla ve vertikální ose (nahoru, dolů).
- **Zatáčení (yaw)** – pohyb letadla kolem kolmé osy, vedoucí letadlem shora dolů. Tento parametr ukazuje odchylku letu od zadaného kurzu. Pohybem kolem kolmé osy měníme vybočení nosu letadla doleva nebo doprava.
- **Klonění (roll)** – pohyb letadla kolem podélné osy, která vede od nosu k ocasu letadla. Kloněním měníme orientaci křídel letadla (jedno křídlo stoupá nahoru, druhé klesá dolů). [3].

Pomocí těchto tří pohybů můžeme popsat polohu letadla ve vzdušném prostoru. Pro automatické řízení letadla tedy musí existovat jednoduchý způsob získávání těchto tří základních úhlů. Takový způsob existuje, je založený na vlastnostech rychle se otáčejícího předmětu – gyroskopu, který zachovává neměnnou polohu své osy v prostoru.

2.2 Gyroskop

Gyroskop (obrázek 2.2) je zařízení, určené k navigaci a určování směru. Princip gyroskopu je znám už od roku 1817, kdy jej objevil Johann Bohnenberger. Jako konstruktér gyroskopu je však označován až Léon Foucault, který na skutečném modelu předváděl v roce 1852 důkaz rotace Země kolem své osy [4].



Obrázek 2.2: Tříosý gyroskop [5]

Základním principem gyroskopu je využití fyzikálních vlastností rychle se otáčejících předmětů. Nejjednodušším gyroskopem je známá dětská hračka káča. Gyroskopy používané v letectví fungují na stejném principu, ale mají mnohem sofistikovanější tvar.

2.2.1 Princip gyroskopu

Setrvačník (rychle rotující předmět) je uložen v tzv. kardanovém závěsu. Kardanův závěs je soustava tří v sobě umístěných kovových prstenců, které jsou spojeny otočnými čepy takovým způsobem, že osy čepů sousedících prstenců jsou na sebe navzájem kolmé. Takový gyroskop může vykonávat nezávislý pohyb kolem všech tří os, protínajících se ve středu závěsu, který zůstává stále nehybný vůči svému základu.

2.2.2 Gyroskopický efekt

Jednou z vlastností gyroskopu, kterou můžeme výhodně využít pro sestrojení autopilota je gyroskopický efekt. Gyroskopický efekt vzniká díky rychle se otáčejícímu setrvačníku v gyroskopu, který klade odpor proti jakékoliv změně své rotační osy a bez působení jiných vnějších sil si jeho osa otáčení

udržuje stále stejný směr. Velikost síly působící proti změně je přímo úměrná velikosti průměru setrvačníku a rychlosti jeho otáček – čím větší je průměr setrvačníku a rychlost jeho otáčení, tím větší a silnější je gyroskopický efekt.

Jestliže například byla osa setrvačníku na začátku směrována na nějakou hvězdu, tak při jakékoliv změně polohy přístroje (tedy i změně polohy letounu), bude osa stále ukazovat na původně zvolenou hvězdu, i když se změní orientace gyroskopu vůči zemským osám.

Gyroskopický efekt poprvé objevil francouzský fyzik Jean Bernard Léon Foucault. V roce 1852 využil této fyzikální vlastnosti pro experimentální důkaz otáčení Země kolem své osy a sestrojil známé Foucaultovo kyvadlo. Díky tomuto experimentu vznikl i název slova „gyroskop“, což v překladu z řečtiny znamená „pozorovat otáčení“.

2.2.3 Precese gyroskopu

Druhou důležitou vlastnost rotujícího gyroskopu nejlépe poznáme, pokud na něj začneme působit další vnější silou. Každou vnější sílu působící na rotující předmět můžeme rozložit na dvojici sil – sílu rovnoběžnou na osu rotace předmětu a sílu kolmou na osu rotace. Síla kolmá s osou rotace rychlost rotace buď urychluje (pokud síla působí po směru rotace) nebo zpomaluje (pokud síla působí proti směru rotace). Při zpomalování rotace gyroskopu nebo vlivem působení vnější síly na gyroskop se začne odchylovat osa rotace tělesa od počátečního směru. To přivádí gyroskop k rotaci kolem vertikální osy [4].

Tento jev se nazývá precese gyroskopu – pohyb rotujícího tělesa kolem volné osy, která opisuje tvar kužele. Rychlost precese nejvíce ovlivňuje rychlost otáčení setrvačníku – čím rychleji se setrvačník otáčí kolem své osy, tím pomalejší bude precese. Pokud na setrvačník přestane působit vnější síla způsobující precesi, tak dojde k okamžitému návratu setrvačníku do původní polohy a precese končí.

Na těchto vlastnostech gyroskopu je založeno několik nástrojů, které se používají pro automatické pilotování letadla a jeho stabilizaci ve vzdušném prostoru.

2.3 První autopilot

První automatický stabilizátor letu byl vytvořen Lawrencem Sperrym v roce 1912. Pro tento systém Sperry použil čtyři gyroskopy pro stabilizaci referenční roviny. S pomocí elektrické, mechanické a pneumatické složky umožňoval Sperryho gyrokompas detekovat pozici letadla relativně k referenční rovině a provádět signály pro korekci polohy. Systém byl sestaven tak, aby poskytoval přibližnou reakci na skokovou změnu polohy letadla. Speeryho systém byl založen na intuitivním chování letadla, nebyl založen na žádných teoretických základech [6].

V létě roku 1914 se v Paříži konal mezinárodní průmyslový veletrh, na kterém Sperry veřejnosti předvedl první systém pro stabilizaci letadla. Spolu s mechanikem svůj nový vynález prezentovali krátkou leteckou ukázkou. Letadlo vzlétlo a udělalo okružní let nad výstavištěm, během kterého vyšel

Obrázek 2.3: Autopilot Wileyho Posta [7]

na křídlo letadla mechanik. Lawrence Sperry, který řídil letadlo pustil řízení a dal ruce nahoru, tak aby návštěvníci veletrhu mohli vidět, že letadlo letí rovně samo i bez pilotova zásahu do řízení a ani mechanik, stojící na křídle, nemůže ovlivnit jeho rovnováhu [7].

Francouzská armáda měla ihned po skončení zkušební demonstrace velký zájem o vynalezený systém. Tento systém Sperryho firma dále rozvíjela a vylepšovala a v 20. letech vytvořila již skutečný autopilot. Tento autopilot Wiley Post v roce 1933 použil pro první samostatný let kolem světa, který zvládl vykonat za méně než 8 dní. Fotografie tohoto autopilota je na obrázku 2.3[8].

Už od poloviny 30. let používalo velké množství leteckých dopravních společností Sperryho systém autopilota pro lety na dlouhé vzdálenosti. V rámci dalšího vývoje autopilota byli zlepšeny algoritmy řízení a hydraulické servomechanismy. Díky rozšíření autopilota o další dodatečné nástroje, zejména o radionavigaci bylo možné uskutečňovat lety i během noci a ve špatném počasí [9].

Během 2. světové války se znovu zvýšily požadavky v letectví (zejména kvůli velkým bombardérům vykonávajícím několikahodinové lety na dlouhé vzdálenosti). To vedlo k nutnosti vypracovat dokonalejší autopiloty. Další zvyšování požadavků na bezpečnost a regularitu letů vynucovalo automatizaci dalších leteckých činností a náročnějších částí letu – jako je příprava k přistání, přistání a vzlet.

2.4 Moderní autopiloty

Moderní systémy autopilota mají mnohem složitější konstrukci, kromě stabilizace letadla v prostoru a udržování kurzu letu umožňují realizovat programované ovládání v různých fázích letu: vzlet, stoupání do letové hladiny, udržování letadla v letové hladině, klesání, přiblížení na přistání a přistání. Nejsložitější systémy automatického ovládání na sebe berou značnou část ovládacích funkcí i v ručním režimu a dělají ovládání pro pilota jednodušším – zmírňují účinky turbulence, zabraňují unášení větrem, pomáhají řešit kritické situace letu a dokonce zakazují nebo ignorují některé činnosti pilota.

Stejně jako pro ovládání letadla jsou autopiloti nezbytní v leteckém průmyslu, protože umožňují snižování nákladů letecké dopravy a zvyšování její bezpečnosti. Let letadla na autopilot spotřebovává výrazně méně paliva než let řízený pilotem, také dochází k menšímu zatěžování konstrukce letadla během letu, vzletu i přistání. Moderní automatické systémy obsahují komplikované součásti, které jsou schopny rozpoznat a adekvátně zareagovat na hrozící kolizi s jinými objekty, mohou také umožnit přistát v situacích špatné nebo žádné viditelnosti [10].

Současné systémy používají počítačové programy pro řízení letadla. Program čte aktuální pozici letadla a získává data z celé sítě senzorů rozmístěných v letadle, na základě těchto vstupních informací potom ovládá zařízení letadla a systém řízení letu. Kromě klasické letové kontroly jsou systémy také schopny ovládat řízení tahu motoru pro optimalizaci rychlosti, dále přesun paliva v nádržích tak, aby došlo k optimálnímu vyvážení letounu ve vzduchu.

Aktuální pozice letadla v prostoru, jeho letová hladina a další informace o pohybu letadla získává autopilot pomocí inerciálního navigačního systému, který používá gyroskopy a akcelerometry pro měření zrychlení ve všech 3 osách. Z těchto měření se následně vypočítává pozice letounu.

2.5 Základní veličiny

2.5.1 Rychlost

Správná rychlost letu je důležitá pro bezpečnost celého letu – let letadla v rychlostech nižších než je dovolená minimální rychlost přivádí letadlo ke ztrátě stability a ovládání. Naopak zvyšování rychlosti letu nad povolenou rychlost je spojeno s nebezpečím zničení letadla [11]. Rozlišuje se vzdušná a traťová rychlost, které se měří v km/h.

- **Skutečná vzdušná rychlost** – rychlost letadla vůči okolnímu vzdušnému prostředí. Tuto rychlost letadlo získává vlivem tažné síly motorů. Vzdušná rychlost závisí na aerodynamických charakteristikách letadla, jeho váze a hustotě vzduchu. Vítr nemá vliv na její hodnotu a směr.
- **Traťová rychlost** – rychlost letadla vůči povrchu země. Její velikost ovlivňuje vítr, který zmenšuje nebo zvětšuje rychlost pohybu letounu vůči zemskému povrchu.

Skutečná vzdušná rychlost, stejně jako traťová jsou rychlosti v přímém fyzikálním smyslu. Přístroje pro jejich bezprostřední měření však v současné době nejsou vytvořeny, proto je potřeba skutečnou vzdušnou rychlost vypočítávat nepřímým způsobem pomocí rychloměrů.

- **Přístrojová rychlost** není rychlostí pohybu letadla vůči jinému objektu na zemi, to znamená že ve fyzikálním smyslu toto vůbec není rychlost. Přístrojová rychlost je v podstatě veličina dynamického tlaku (tj. tlaku vytvořeném rychlostí vzdušného proudění okolo letadla), vyjádřená v rychlostních jednotkách. Přístrojová rychlost se měří pomocí Pitotovy nebo Venturiho trubice, které jsou umístěny v přední části letadla nebo na křídlech tak, aby byly pokud možno v neovlivněném proudu vzduchu. Tyto trubice neměří rychlost letu přímo, ale pouze měří tlak vyvolávaný tlakem proudícího vzduchu. Na rychlost je tento změřený tlak nutné převést. Přístrojová rychlost, kterou pilot čte z rychloměru, je zatížená několika chybami - přístrojové, aerodynamické a metodické. Z tohoto důvodu má pilot v letadle k dispozici hned několik různých rychlostí. [1].
- **Indikovaná rychlost letu (IAS)** – je to údaj z rychloměru vestavěného v letadle opravený pouze o přístrojovou chybu, způsobenou konstrukcí rychloměru. Tato chyba je však běžně zanedbatelně malá, proto IAS prakticky představuje přístrojovou rychlost letu, kterou čte pilot z rychloměru.

2.5.2 Kurz letu

Kurz letu má velký význam pro navigaci, protože je zároveň pilotážním a navigačním elementem. Kurz ukazuje, kam směřuje podélná osa letadla. Vítr ovlivňuje směr letu, proto kurz letadla se shoduje se skutečným směrem jeho pohybu vůči Zemi pouze v bezvětří. Ve větru může být směr podélné osy letadla jiný než kurz letu [12].

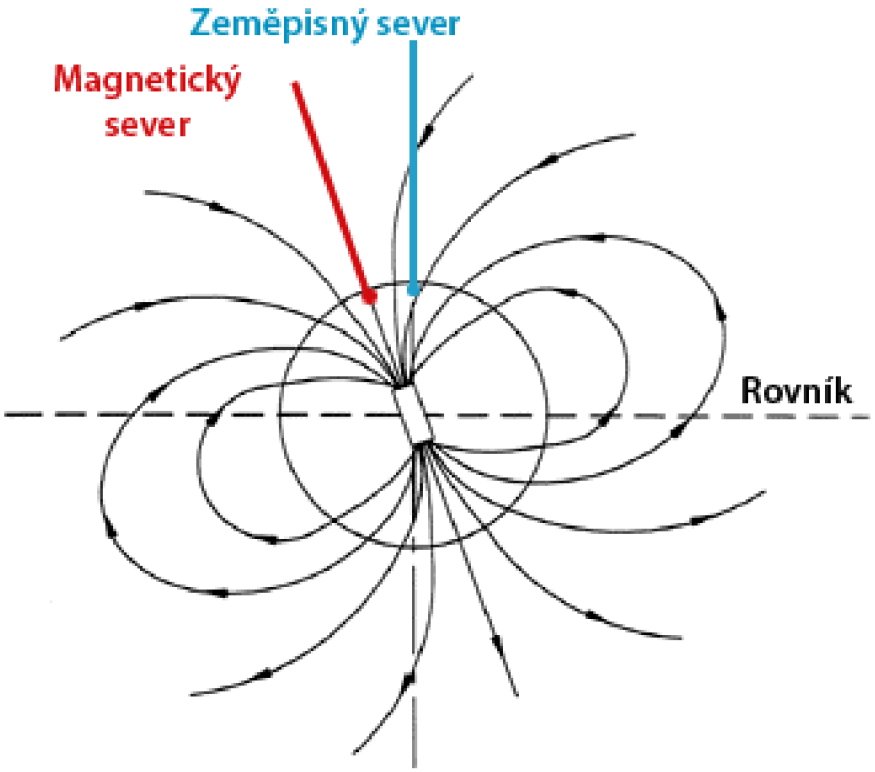
Existují dva způsoby měření kurzu – zeměpisný a magnetický. Pro pochopení jejich rozdílu je nutné nedříve definovat pojem zeměpisný a magnetický poledník.

Zeměpisný poledník

Proložíme-li rovinu zemskou osou, dostaneme na obvodu zeměkoule, kde tato rovina protíná její povrch, myšlenou kružnici, kterou nazýváme poledníková kružnice. Polovina této kružnice je poledník. Poledník je nekratším spojem polů na Zemi, a proto určuje zeměpisný sever a jih. Základní nultý poledník prochazí londýnskou hvězdárnou v Greenwich.

Magnetický poledník

Projekce siločar geomagnetického pole na povrchu Země. Jsou to složité křivky, shodují se na severním a jižním magnetickém poli Země.



Obrázek 2.4: Magnetické póly Země [13]

Zeměpisný kurz

Zeměpisný kurz je úhel, který svírá směr zeměpisného severu daného poledníkem v místě výpočtu se směrem projekce podelné osy letadla na horizontalní plochu. Měříme ho od poledníku ve směru pohybu hodinových ručiček ve stupních od 0° do 360°. V podstatě kurz ukazuje směr nosu letadla vůči zeměpisnému severu. Číselný údaj musí podle pravidel mít vždy právě tři cifry. Kurz 000 ukazuje směr objektu na sever, 090 na východ, 180 na jih, 270 na západ.

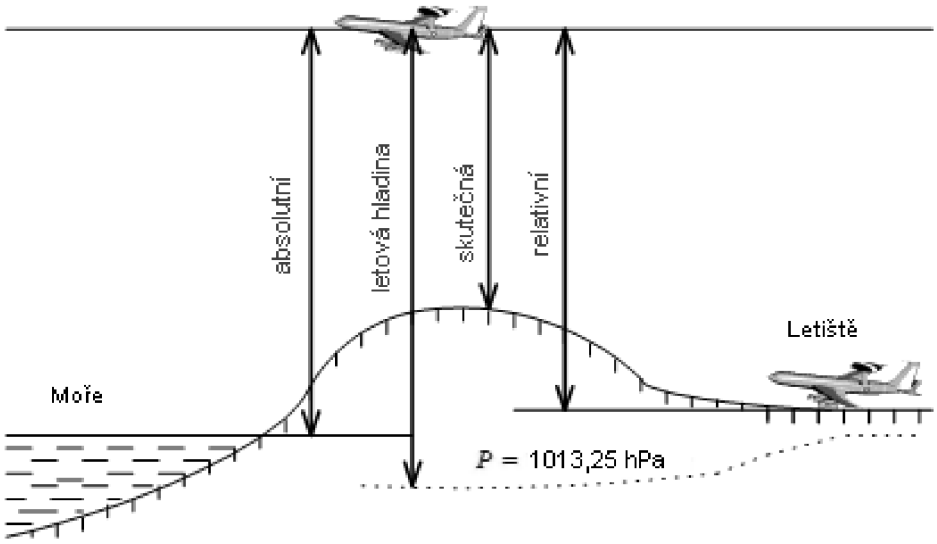
Magnetický kurz

Polohy magnetických pólů Země, ke kterým směřují střelky kompasů určujících magnetický kurz, nejsou totožné s polohami zeměpisných pólů zeměkoule. Vzdálenost zemských magnetických pólů od zeměpisných pólu je přibližně 2000 km. Poloha magnetických pólů na zeměkouli navíc není stálá – mění se s časem.

Magnetický kurz je úhel, který svírá směr magnetického severu daného magnetickým poledníkem v místě výpočtu se směrem projekce podélné osy letadla na lokální horizontální plochu. Měříme od magnetického poledníku ve směru pohybu hodinových ručiček, ve stupních od 0° do 360°. Zeměpisné směry nelze v zásadě slučovat se směry magnetickými, jak ukazuje i obrázek 2.4.

2.5.3 Výška

Pro jednoznačnou definici polohy letadla je nutno kromě zeměpisných souřadnic znát také správnou výšku letu. Výškou letu se nazývá horizontalní vzdálenost letadla od počáteční úrovně. Znalost výšky je nezbytná pro udržování režimu letu, řízení letadla a zajištění bezpečnosti letu. Výška letu se měří v metrech nebo ve stopách (feet).



Obrázek 2.5: Klasifikace výšky letu podle počáteční úrovně [14]

Klasifikace vyšky letu podle počateční urovně (obrázek 2.5):

- **skutečná** – odpočítávaná od úrovně místa, do kterého letíme,
- **relativní** – odpočítávaná od nějaké dohodnuté úrovně, například letiště,
- **výška letové hladiny** – odpočítávaná od úrovně s tlakem 1013,25 hPa,
- **absolutní vyšku** – odpočítávaná od úrovně moře.

Podle podmínek letu rozlišujeme výšky velmi malé (do 200 m), malé (200-1000 m), střední (1000-4000 m), velké (4000-12000 m) a stratosferické (více než 12000 m).

2.6 Požadavky na autopilota

Hlavní funkcí autopilota a jeho hlavní úlohou při automatickém pilotování je správné udržování orientace letadla pohybujícího se ve vzdušném prostoru. Díky stabilizaci polohy letounu ve vzduchu nedochází k odchylování od zadaného kurzu, protože trajektorie letu závisí na správné orientaci letadla.

Autopilot udržuje letadlo v zadaném režimu letu bez zásahu pilota. Proto je nutné, aby parametry, které charakterizují daný režim letu, zůstávaly neměnné nebo alespoň jejich odchylování bylo co nejmenší. Kromě toho je potřeba odstraňovat kmitání letadla kolem všech jeho tří os.

Některé parametry, charakterizující stav rovnováhy letadla vůči jeho osám, se mohou měnit během letu (vyvážení, hmotnost, tah motoru), autopilot musí bez zásahu pilota udržovat letadlo v zadané poloze i přes tyto dynamické změny probíhající za letu. Autopilot musí vykonávat všechny základní letecké manévry – přímý let v letové hladině, stoupání či klesání do nové letové hladiny, pravé a levé zatáčky a také umět provést automatické přistání a vzlet.

Současná stabilizace čtyř základních parametrů letu – rychlosti, výšky, kurzu a náklonů je možná s využitím čtyř základních ovládacích prvků letadla – tahu motoru, výškovky, směrovky a křidélek. Autopilot proto musí tyto čtyři základní prvky letadla obsluhovat [15].

2.7 Problémy systémů automatického řízení letu

Základním problémem při konstruování autopilotů a automatických systémů řízení je bezpečnost letu. V jednoduchých autopilotech je předpokládána velká kontrola pilotem – v případě nestandardní situace musí dojít k rychlému odpojení autopilotního systému pilotem. Podobně při jakékoliv poruše jeho normálního fungování je zvolen okamžitý přechod na ruční ovládání letounu.

U složitějších autopilotů se však již od počátků vývoje klade velký důraz na spolehlivost a zachování funkčnosti i v kritických situacích. Předpokládají se opatření pro zvýšení bezpečnosti letu. Systémy jsou mnohokanálové, tj. současně pracují 2, 3 výjimečně i 4 stejné kanály řízení. Výpadek jednoho nebo dvou kanálů žádným způsobem neovlivňuje funkčnost systému jako celku. V případě vzniku výpadku systém samostatně rozhoduje o možnostech zachování činnosti autopilota, přepnutí na rezervní kanál nebo předání řízení pilotu.

Vzhledem v celkové složitosti automatických systémů řízení existuje možnost, že některá část systému může selhat. Byly dokonce případy, kdy letadlo havarovalo právě kvůli selhání systému autopilota. Dlouhodobé záznamy ale ukazují, že k havárii, která je způsobena poruchou autopilota, typicky došlo z důvodu, že pilot neodpojil autopilot před pokusem o manuální ovládání. Většinou se tedy nejedná o fatální selhání autopilotního systému, ale o selhání komunikace mezi pilotem a tímto systémem, jeho špatné pochopení či nesprávné použití.

Kapitola 3

Interakce pilota se systémem automatického pilotování

Pro návrh správného grafického rozhraní je potřeba brát v úvahu typické způsoby interakce mezi programem a jeho uživatelem. Informace poskytovaná programem a zobrazení této informace na displeji musí umožňovat uživateli provádět svou činnost úspěšně a spolehlivě.

Již od vzniku prvních systémů pro automatické řízení letu byli vynálezci znepokojeni jejich problémy a zejména problémy s interakcí mezi pilotem a autopilotním systémem. Ve zprávách o leteckých nehodách a poruchách se neustále objevovaly dva faktory, které byly za tyto nehody zodpovědné – rozhraní poskytovalo autopilotu nedostatečné informace o stavu stroje, pilot měl nesprávné představy o fungování a chování stroje. Oba tyto faktory přiváděly posádku letadla k neočekávanému chování, zmatku a chybám [16]. Nicméně viníky těchto chyb spolupráce mezi autopiloty a piloty nelze často jednoznačně označit. Nelze jednoznačně rozhodnout, zda za tyto chyby bylo zodpovědné pouze rozhraní autopilota a jeho interakce s pilotem nebo zda byla chyba způsobena lidskou chybou, nedostatečným školením, špatným uvědoměním si krizové situace nebo nesprávně zvoleným postupem.

V interakci mezi pilotem a autopilotem hrají klíčovou roli tři prvky. Tyto tři prvky spolu zajišťují korektní a spolehlivou interakci mezi uživatelem a systémem:

1. množina úkonů, které musí pilot provést pro ovládání nástrojem,
2. uživatelský model chování nástrojů, který pilot zná a má naučený z trénování na simulátoru a z manuálů,
3. tozhraní, pomocí kterého pilot získává informace o chovaní nástrojů.

3.1 Obecné zásady

Z důvodu velkého množství různých informačních panelů v kokpitu letadla a jejich složitosti je potřeba dodržovat zásady správného zobrazování informací tak, aby se povedlo co nejvíce zjednodušit a zabezpečit interakci pilota s programem. Americké vědecké studie ukazují, že dobré grafické rozhraní s leteckou informací může výrazně zmenšit problémy s chybami a leteckým incidenty. Výsledkem těchto studií je několik obecných zásad a doporučení [17]:

1. zobrazovat letovou informaci co nejjednodušším a nejpreciznějším způsobem,
2. navrhovat ovládací nástroje autopilota tak, aby se minimalizoval počet kroků, které musí pilot vykonat pro požadovanou akci,
3. všechny rutinní, nekritické informace musí být úplně zautomatizované tak, aby zbytečně nezatěžovaly a nekomplikovaly konfiguraci autopilota,
4. odstranit všechny nepodstatné a zbytečné operace na ovládací konzoli autopilota,
5. symboly a zprávy musí být logicky umístěny a musí pracovat se souvisejícími informacemi,
6. pro zvýraznění informace nepoužívat pouze změnu barvy, informace musí být identifikovatelná minimálně pomocí dvou různých parametrů (například rozměr, tvar, barva, umístění),
7. varovné informace se musí objevovat na stejném místě ve kterém jsou za normálních okolností běžná data.

3.2 Touch screen displej

Touch screen technologie se v dnešní době uplatňují v široké škále různých informačních systémů a aplikací. Jejich použití je zejména vhodné v systémech obsluhy zařízení – prodejní terminály, samoobslužné automaty a podobně. Pro autopilota v letadle lze také využít výhod dotykových displejů.

Výhody

- Přímočarost – na rozdíl od nepřímých způsobů ovládání (myš, joystik, klávesnice) uživatel přímo komununikuje s objektem, který jej zajímá. Není třeba vyhledávat vstupní zařízení, pomocí kterého nejdříve vybereme požadovaný objekt a ztrácet vizuální soustředěni na objekt. Stejně tak uživatel nemusí mapovat ruční pohyb na pohyb kurzorem, jak to vyžaduje klasické ovládání například pomocí myši či joystiku.
- Rychlost – pro mnoho úkolů je ovládání pomocí touch screen rychlejší, protože uživatel nemusí vyhledávat vstupní zařízení, může přímo vybrat požadovaný objekt.

- Jednodušší učení – rychlejší pro pochopení. Uživatelská rozhraní pro dotykové displeje mívají jednoduchou strukturu a nepožadují složité ovládání pomocí myši.
- Flexibilita – touch screen podporuje možnosti, které nejsou dosažitelné pomocí klávesnice. Uživatel si může vybrat, jaké rozložení klávesnice preferuje, preferovaná klávesnice se poté zobrazuje na displeji.
- Nepotřebuje žádné dodatečné místo – není potřeba žádných dalších vstupních zařízení, postačuje pouze displej.

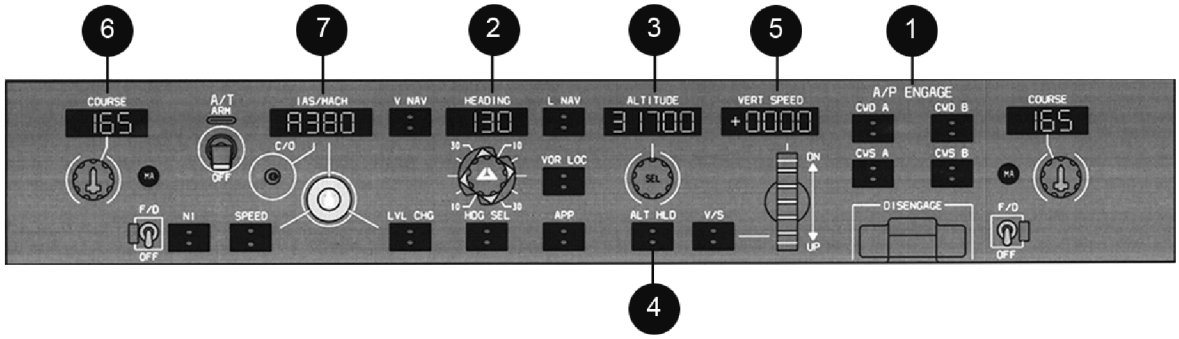
Nevýhody

- Není haptická (hmatová) odezva. Při ovládání klasických tlačítek je možné cítit spoustu informací i pomocí hmatu – je tak například možné rozeznat zmáčknuté tlačítko od nezmáčknutého, poznat konec rozsahu u otočných voleb a podobně.
- Při ovládání vertikálního displeje musí uživatel držet ruku ve vzduchu, proto jsou vertikální displeje vhodné pouze pro epizodickou práci (například pro bankomaty). Při dlouhodobějším ovládání displeje by hrozila nadměrná únava uživatele systému.
- Při ovládání horizontální displeje omezují ruce, které ovládají displej, rozhled na zbývající části displeje. Pokud ovládáme prvky v jedné části displeje, můžeme si zároveň zastiňovat informace zobrazované v druhé části.
- Paralax – omezuje přesnost pozicování úkonů uživatele na dipleji bez kurzoru. Použití kurzoru ale přidává dodatečnou složitost, která zmenšuje úspornost celého řešení.

Kapitola 4

Analýza existujících autopilotů

Pro lepší pochopení systémů automatického pilotovaní se podíváme na autopilot ve skutečném letadle Boeing 737 (obrázek 4.1). Zaměříme se pouze na hlavní funkce autopilota.



Obrázek 4.1: Rozhraní autopilota Boeing 737

1. **Autopilot master switch** – zapíná autopilota, udržuje stávající klopení a klopení. Pokud je autopilot zapnut, není možné řídit letadlo, pilot může pouze kontrolovat a monitorovat nastavení autopilota. Jestli jsou spolu se zapnutým autopilotem zapnuty režimy Heading mode, Altitude mode nebo Navigation mode, letadlo začne ihned vykonávat aktivované příkazy. Proto je dobré před zapnutím autopilota zkontrolovat jeho nastavení, zejména aktivaci funkcí.

2. **Heading selection window/bug** – umožňuje nastavit Heading mode. Pro aktivaci režimu je potřeba zmáčknout tlačítko HDG SEL, hned potom letadlo přijme požadovaný kurz. Nastaveni HDG ukazuje magnetický kurz.

3. **Altitude selection window** – nastavení letové hladiny. Letadlo začne klesat nebo stoupat na nastavenou výšku.

4. **Altitude hold mode selector button** – po zapnutí je aktivován režim Altitude hold. To znamená, že autopilot bude udržovat stávající výšku letu.

5. **Vertical speed selection window** – nastavení vertikální rychlosti. Umožňuje nastavit rychlost, kterou bude letadlo klesat nebo stoupat.

6. **Course selection window/knob** – umožňuje nastavit kurz letu, který odpovídá zeměpisnému kurzu.

7. **Airspeed/Mach selection window** – nastavení letové rychlosti.

Kapitola 5

Aplikace

5.1 Specifikace požadavků

V první části práce byli prozkoumány teoretické základy systémů pro automatické řízení letadla, obecné požadavky kladené na systém autopilota, časté problémy spojené s obsluhou tradičních rozhraní a také základní zásady používané při navrhování grafického rozhraní pro zobrazování informací a řízení letadla. Pomocí těchto informací lze provést specifikaci požadavků na nově navrhované grafické rozhraní. Aplikace by měla odpovídat potřebám pro přehledné zobrazení informací a umožňovat pohodlnou a jednoduchou interakci s uživatelem pomocí touch screen displeje. Nově vyvinutá aplikace by měla být postavena na grafickém rozhraní Qt a implementována v jazykovém prostředí C++.

Aplikace by měla umožňovat výstup a příjem dat pomocí protokolu vyvíjeného v prostředí laboratoře SimStar a jako rozšíření také pomocí protokolu CANaerospace, který je používán pro letecké aplikace. Výsledná aplikace bude používána v simulátoru v laboratoři SimStar.

5.1.1 Funkční požadavky

Musí být umožněno přepínání jednotek jednotlivých veličin mezi metrickou SI soustavou a imperiální soustavou. Každá změna hodnoty musí být v určitém časovém intervalu potvrzena, jinak dojde k návratu na původní hodnotu před změnou. Autopilot lze zapnout až po potvrzení všech nastavených hodnot.
Rozhraní autopilota umožňuje ovládání pěti základních veličin:

- **rychlost (IAS)** – je zobrazena v jednotkách knots nebo v mph. Pilot může nastavovat rychlost pouze v určitém intervalu, který je daný aktuální polohou klapek, jak ukazuje tabulka 5.1.
- **výška (ALT)** – uváděná v jednotkách feet nebo metrech. Pilot může aktivovat funkci udržování aktuální letové výšky pomoci tlačítka ALT HOLD.
- **vertikální rychlost (VS)** – uváděná v jednotkách ft/min nebo m/s.

- **magnetický kurz (HDG)** – uváděný ve stupních.
- **zeměpisný kurz (TRK)** – uváděný ve stupních.

Rozsah nastavování rychlosti je závislý na poloze klapek, proto je pro tuto veličinu samostatná tabulka 5.1.

Veličina	Zkratka	Jednotky	Klapky	Minimum	Maximum
Rychlost	IAS	uzly	Zasunuty	37	146
			Vysunuty	37	70
		km/h	Zasunuty	68	270
			Vysunuty	68	130

Tabulka 5.1: Specifikace rychlosti

Zbývající veličiny, rozsahy a jejich jednotky jsou zobrazeny v následující tabulce 5.2.

Veličina	Zkratka	Jednotky	Minimum	Maximum
Výška	ALT	feet	0	10000
		m	0	3000
Vertikální rychlost	VS	ft/min	-2000	2000
		m/s	-10	10
Magnetický kurz	HDG	stupně	0	360
Zeměpisný kurz	TRK	stupně	0	360

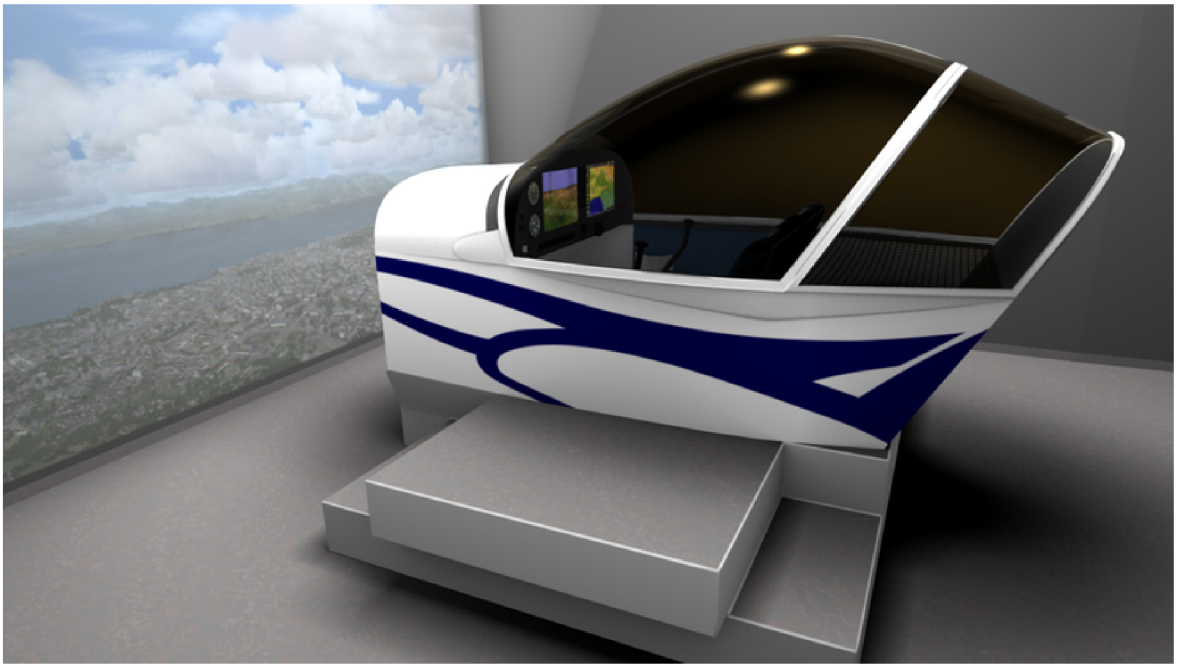
Tabulka 5.2: Specifikace veličin autopilota

5.1.2 Jiné požadavky

- **Jednoduchost a přehlednost** – navrhovaná aplikace by měla být co možná nejjednodušší, aby obsluhující piloti mohli rychle najít a provést akce, které potřebují.
- **Implementace** – systém bude implementován v jazyce C++, s pomocí grafického rozhraní Qt

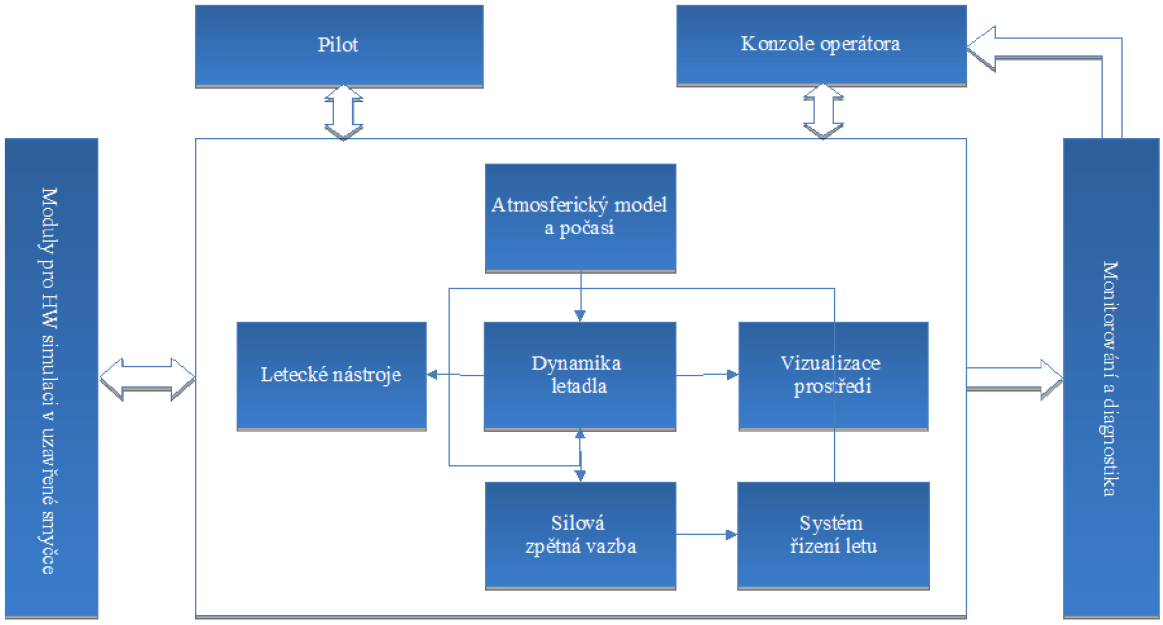
5.2 Laboratoř SimStar

Vyvíjená aplikace je součástí systému v multimediální výzkumné laboratoři SimStar. Laboratoř se zabývá výzkumem bezpečnosti létání a vyvíjí nástroje pro výcvik a trénování pilotů. Laboratoř je vybavena trupem letounu SportStar (obrázek 5.1) a kompletním hardwarovým a softwarovým vybavením pro simulaci letu. V druhé části místnosti se nachází výpočetní vybavení a stanoviště operátora. Výpočty a zobrazování simulačních dat mají na starosti dva počítače, jeden počítač zajišťuje běh simulačního jádra a jeho vizualizaci na projektor, druhý počítač má na starosti zobrazování informací na displejích, které se nacházejí v kabině letounu a na stanovišti operátora. Schéma simulátoru SimStar je zakresleno na obrázku 5.2.



Obrázek 5.1: Simulátor SportStar v laboratoři SimStar [18]

Kabina letounu je vybavena dvěma moderními dotykovými displeji, na kterých se zobrazují informace o průběhu letu. Tyto informační prvky jsou označovány jako hlavní letový displej (PFD – Primary Flight Display).



Obrázek 5.2: Schéma simulátoru SimStar [19]

5.2.1 Hlavní letový displej

Hlavní letový displej je moderní letecký nástroj vyhrazený pro zobrazování leteckých informací. Kombinuje zobrazení šesti hlavní leteckých údajů – rychlost letu, ukazatel kurzu, výška letu, vertikální rychlost, umělý horizont a zatáčkoměr na jednom displeji. Díky tomu má pilot všechny důležité informace přehledně a jednoduše zobrazeny na jednom místě. V případě poruchy hlavního letového displeje však v kokpitu vždy zůstávají i mechanické ukazatele.

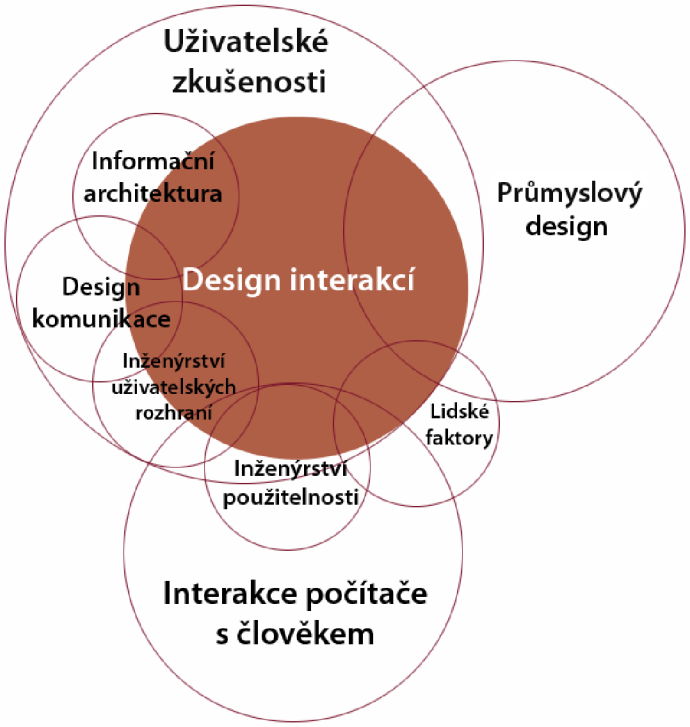
Letový displej v laboratoři SimStar tento koncept dále rozvíjí a zdokonaluje. Na displeji je tak možné kromě šesti základních přístrojů vidět i provozní informace jako jsou otáčky motoru, výkon motoru, množství paliva v palivových nádržích či letovou mini-mapu se souřadnicemi aktuální polohy. Umělý horizont je navíc zobrazen ve skutečném terénu, nad kterým letoun právě prolétá.

Nově vyvíjená aplikace rozhraní autopilota je dalším rozšířením hlavního letového displeje.

Kapitola 6

Návrh aplikace

6.1 Návrh interakce

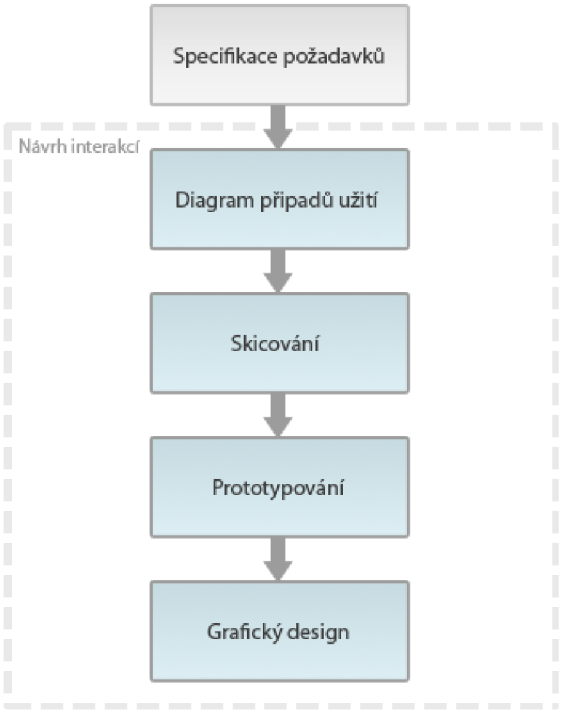


Obrázek 6.1: Diagram závislostí designu interakcí [20]

Slovo interakce původně v latinském překladu znamená jednat mezi sebou. Interakcí v počítačovém designu rozumíme jednání mezi uživatelem aplikace a počítačovým systémem. Design interakce je oblast odborných znalostí zaměřených na návrh chování systémů, se kterými uživatel interaguje. V návrhu interakce nás zajímá celý proces komunikace, ovládání a řízení systému, a také vzájemná

oboustranná aktivita obou interagujících subjektů. Na diagramu 6.1 vidíme vztahy návrhu interakcí (interaction design) s ostatními inženýrskými obory, které ovlivňují návrh interakce. Mezi tyto obory patří zejména – user experience design, průmyslový design, informační architektura, inženýrství uživatelských rozhraní (user interface engineering), interakce počítače s člověkem (human computer interaction). Tyto inženýrské disciplíny mají své specifické úzké zaměření na řešení dílčích úkolů. Proces návrhu interakce v sobě zahrnuje všechny tyto obory a jejich kombinací umožňuje vytvořit kompletní návrh systému.

Návrh interakce se skládá z několika částí, jak ukazuje obrázek 6.2. Pro komplexní, kvalitní a vyvážený návrh je důležité dodržování pořadí vývoje designu aplikace – každá etapa vývoje má svůj přínos pro celkovou kvalitu navrhovaného systému a každá další etapa navazuje na tyto přínosy. Zejména je důležité nevynechávat etapy skicování a prototypování, protože jsou základem procesu návrhu interakce.



Obrázek 6.2: Etapy návrhu uživatelského rozhraní

6.1.1 Diagram případů užití

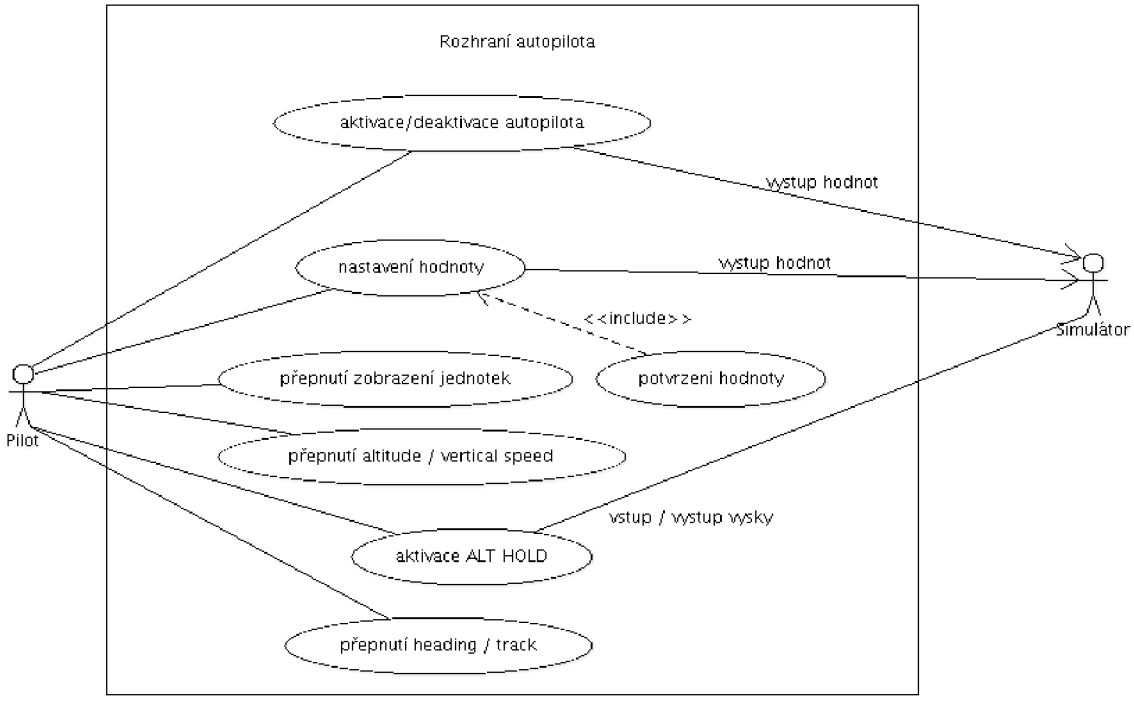
Diagram případů užití je jeden z diagramů, který používá formální modelovací jazyk UML (Unified Modeling Language). Pomocí diagramu můžeme jednoduše popsat vnější pohled na modelovaný

systém a nalézt a vymezit hranici mezi systémem a okolím – definovat rozsah systému. V návrhu interakce nás na diagramu případu užití zajímají interakce mezi uživatelem a systémem.

Uživatelé systému a další vnější objekty, které interagují se systémem, jsou na diagramu případu užití zobrazeny jako figurky. Modelovaný systém je zobrazen jako velký obdélník, v němž jsou obsaženy jednotlivé případy užití systému. Pokud se uživatel účastní případu užití, tak je to na diagramu znázorněno čárou – asociací. Asociace neznamená provedení primitivní funkce, ale znázorňuje komplexní interakci aktéra se systémem, vedenou k určitému cíli.

Diagram případů užití systému autopilota

Se systémem autopilota interagují dva externí aktéři – pilot a simulátor. Pilot je uživatel rozhraní systému autopilota. Role simulátoru zastupuje externí systém (letecký simulátor), se kterým autopilot komunikuje – přebírá aktuální letové hodnoty a vrací nastavení autopilota. Nejvíce používaným případem užití v systému je nastavení hodnoty, kdy uživatel nejdříve musí nastavit hodnotu, potvrdit ji a až poté se změněná hodnota odešle do simulátoru.



Obrázek 6.3: Use case diagram systému autopilota

Specifikace případu užití – nastavování hodnoty

Popis: Pilot provede změnu hodnoty veličiny a odešle data do simulátoru.
Vstupní podmínky: žádné
Výstupní podmínky: Systém nastavil novou hodnotu veličiny.

Scénář užití:

1. OPAKUJ
   (a) Pilot s pomocí posuvníku a tlačítek „+“ a „-“ nastaví novou hodnotu veličiny.
2. Systém po nastavení nové hodnoty očekává její potvrzení.
3. POKUD pilot v předem definovaném časovém intervalu zmáčkne tlačítko „CONF“.
   (a) Dojde k potvrzení nové hodnoty a jejího zapamatování systémem.
   (b) Systém pilotovi zvýrazní tlačítko „CONF“, čímž signalizuje potvrzení nové hodnoty.
   (c) POKUD je autopilot aktivní (tlačítko „AP“ svítí zeleně)
      i. Dojde k odeslání nově nastavené hodnoty do simulátoru.
4. JINAK
   (a) Z důvodu nepotvrzení nové hodnoty systém nově nastavenou hodnotu ignoruje a na displej zobrazí naposledy potvrzenou hodnotu.

6.1.2 Skicování

Proces tvorby uživatelského rozhraní začíná papírovým prototypováním – skicováním. Výhodou této metody návrhu rozhraní je rychlost a jednoduchost tvorby základního konceptu navrhovaného programu. Během této etapy se snažíme definovat potřebnou množinu prvků programu, najít jejich nejoptimálnější rozvržení a umístění v prostoru. Použití papírové metody je pro tyto návrhy nejrychlejším a nejpřirozenějším způsobem. Při použiti digitálních nástrojů v této etapě vývoje může analytikovu pozornost odvádět od konceptu systému k nedůležitým detailům prezentace řešení (například ke grafickým efektům). Naopak použití skicování od těchto detailů oprošťuje a umožňuje se soustředit na strukturu, logiku a koncept vyvijeného programu, nikoliv na jeho grafickou podobu.

6.1.3 Prototypování

Prototypování je proces vytváření prototypu programu, tj. šablony – zkušební verze programu s cílem ověřit použitelnost navrhované koncepce. Prototyp imituje funkce a chování výsledného systému s výrazně nižšími náklady na vývoj než by měla skutečná implementace programu. Tím umožňuje

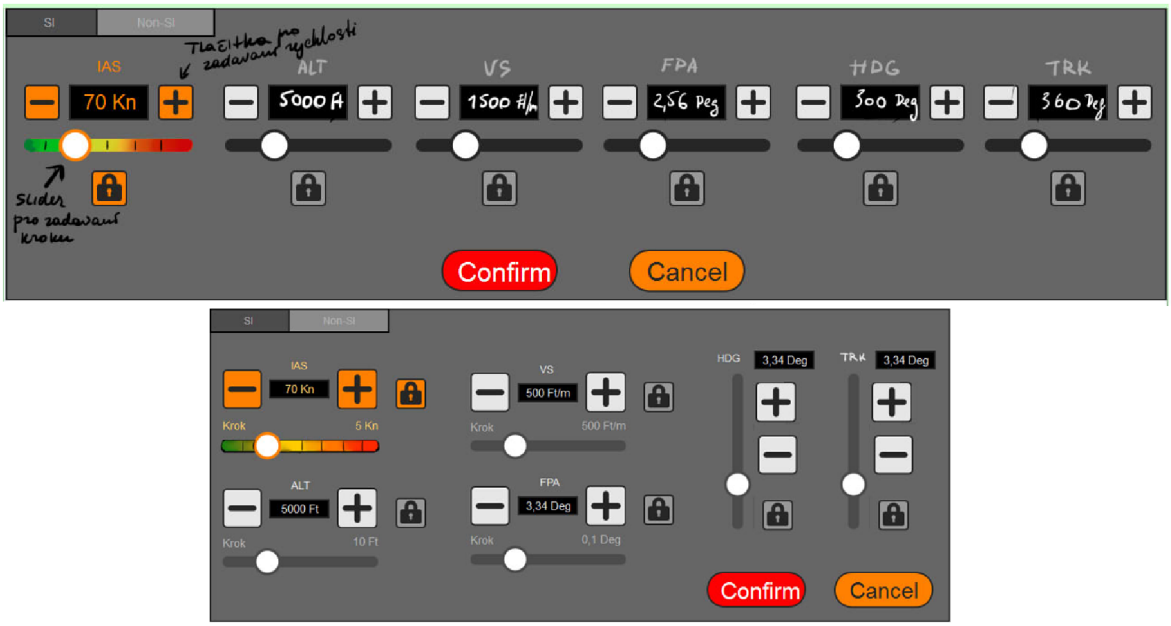
získat zpětnou vazbu od budoucích uživatelů systému dříve než dojde k nákladnému vývoji. Díky nízkým nákladům na vývoj prototypu můžeme navíc provádět iterativní prototypování a postupně tak nalézt optimální uživatelské rozhraní.

Vyvíjená aplikace je určena do prostředí leteckých systémů, které spíše preferují standardizovaná uživatelská rozhraní před novými inovativními. Dalším omezením designu interakce představuje nasazení aplikace na dotykové displeje. Z těchto důvodů je nutné použít řadu standardních ovládacich prvků a dodržovat zásady vývoje uživatelského rozhraní pro dotykové displeje:

- dostatečná velikost ovládacích prvků (dotyková plocha ovládacího prvku nesmí být menší než prst člověka),
- ovládací prvky by neměly být umístěny příliš blízko u sebe a příliš blízko k okraji obrazovky.

V této etapě vývoje během hledání optimálního rozhraní bylo vytvořeno několik prototypů.

První fáze prototypování



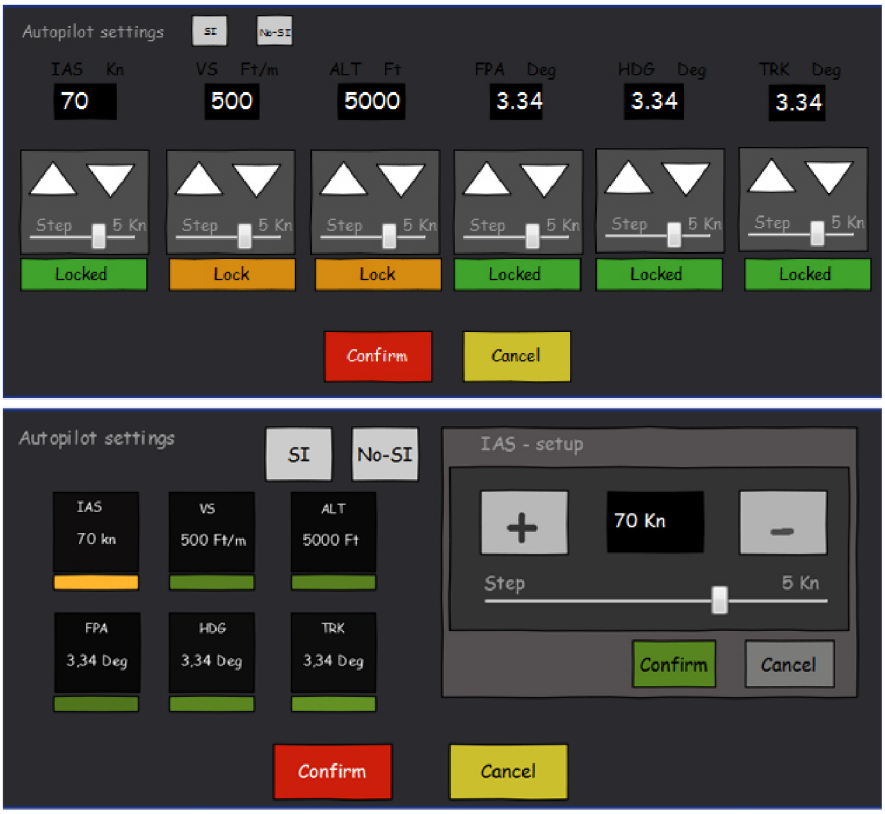
Obrázek 6.4: První a druhý prototyp systému

V prvním prototypu systému (obrázek 6.4) hledám základní orientaci aplikace: vertikální nebo horizontálním. Tlačítka + a − jsou pro změnu velikosti hodnoty veličiny, posuvník nastavuje velikost kroku, po kterém se mění hodnota při stisknutí tlačítek + a −. Tlačítko s obrázkem zámku je pro potvrzení nastavené hodnoty. Ukázalo se, že obě verze jsou nevyhovující – aplikace je příliš široká nebo příliš vysoká pro zobrazení na displeji.

Druhá fáze prototypování

V druhé fázi prototypování (obrázek 6.4) zkouším najít kompaktnější rozvržení aplikace. Oproti předchozímu návrhu zvětšuji velikost tlačítek pro nastavení hodnot tak, abych dodržela zásady vývoje rozhraní pro dotykové displeje. Po této úpravě rozhraní stále obsahuje příliš mnoho prvků a tlačítka pro potvrzení hodnoty nejsou umístěna na spravném místě – vytvářejí vizuální šum.

Třetí fáze prototypování



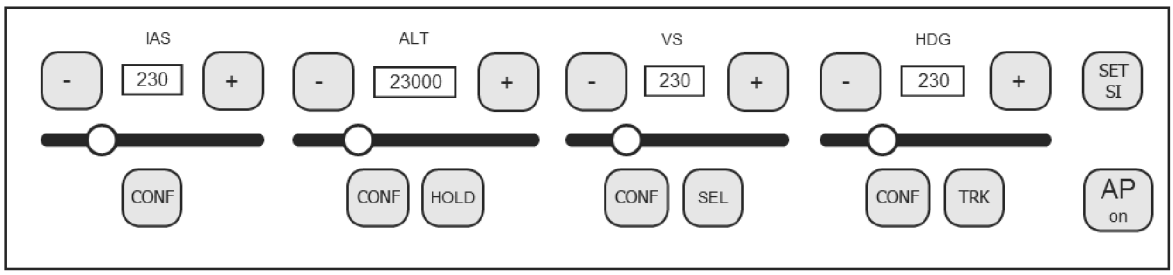
Obrázek 6.5: Třetí a čtvrtý prototyp systému

Hledám nová řešení, která pomohou rozmístit prvky přehledně na malé obrazovce (obrázek 6.5), a zároveň se snažím zmenšit počet ovládacích prvků rozhraní. Stále je ale zobrazeno příliš mnoho zbytečných informací, které bude potřeba dále redukovat.

Poslední fáze prototypování

Výsledkem fáze prototypování (obrázek 6.6) je konečný návrh uživatelského rozhraní. Byly odstraněny zbytečné informace: informace o používaných jednotkách, nastavování kroku. Protože se

aplikací používá vždy pouze jedna z hodnot kurzu – buď pouze magnetický kurz nebo pouze zeměpisný kurz, tak bylo zvoleno rozmístění hodnot `HDG` a `TRK` na stejné místo. Pro přepínání mezi nimi je zavedeno nové tlačítko `SEL`. Byla optimalizována funkcionalita nastavování hodnot: posuvník slouží pro rychlejší změnu hodnoty ve větších krocích, naopak operační tlačítka mají menší krok a slouží pro zpřesňování hodnoty. Krok nastavovaní je pevný pro každou veličinu.



Obrázek 6.6: Finální prototyp systému

6.1.4 Vizuální návrh

Ve vývoji grafického rozhraní hraje vizuální návrh základní roli. Komunikace mezi programem a člověkem probíhá vizuálně. Dobrý grafický design programu působí na člověka pozitivně a v případě GUI desktopové aplikace je zodpovědný za celkový dojem z programu. Správně navržený grafický design zjednodušuje proces interakce mezi programem a člověkem a může výrazně snížit počet uživatelských chyb a únavu obsluhujícího člověka. Návrh vizuálního stylu rozhraní by se měl řídit základními pravidly typografie a grafického designu:

- **používat pouze jeden vizuální jazyk** – všechny části designu musí být provázané mezi sebou,
- **omezit počet typů písma** – používat maximálně 3 typy,
- **musí mít akcent** – soustředit pohled uživatele na jednu hlavní věc pomocí volby správného kontrastu, barvy či typu písma. Pro každý další element informace postupně snižovat úroveň zvýraznění od akcentu k obyčejné informaci. To pomáhá uživateli se postupně zorientovat v aplikaci,
- **barva musí vyvolávat správné emoce** – především musí být vhodné pro druh aplikace,
- **kontrast a rytmus** – seskupovat bloky obsahující příbuzné infrormace, rozmísťovat bloky blíž či dál od sebe, zmenšovat nebo zvětšovat bloky informací a tímto způsobem vytvářet kontrast a rytmus.

Vizuální návrh rozhraní autopilota

Pro navrhovanou aplikaci bylo rozhodnuto modernizovat klasický vizuální styl používaný na stávajících rozhraních autopilotů. Nově navrhované rozhraní v sobě bude sdružovat klasické i moderní prvky designu.
Design by v sobě měl zohledňovat následující klasické elementy:

- neutrální paleta šedo-modrých barev tmavšího odstínu,
- držet akcent na zobrazovaných hodnotách,
- zdůrazňovat kontrast pozadí a operačních prvků,
- světlé barvy pro písmo.

Modernizovaným elementem designu je vyhýbání se hranatým a plochým prvkům.

Vzhledem k tomu, že aplikace bude běžet na dotykových displejích, ve kterých chybí hmatová odezva, bylo navíc jedním z úkolů grafického designu bylo co nejvíce simulovat objem ovládacích prvků programu. Toho bylo dosáženo pomocí gradientu, světla a stínu a také pomocí správné grafické odezvy na stisknutí a puštění tlačítek.

Pro zvýrazňování aktuálně důležitých tlačítek jsou použité barvy: žlutá barva pro stav potvrzovacích tlačítek, zelená pro aktivační stav tlačítek `HOLD` a `AP` (aktivační tlačítko autopilota). Tlačítko `AP` je navíc zvýrazněno světlým rámečkem kolem tlačítka.

Výběr neutrální barvy a celkově tmavé palety byl řízen doporučenými vlastnostmi uživatelského rozhraní pro letecké aplikace. V kabině pilota se nachází velké množství různých nástrojů zobrazujících důležité informace o stavu letadla a také velké množství informací zobrazujících běžné provozní informace. Je důležité dodržovat pravidla pro tvorbu designu pro letecké prostředí, proto bylo nežádoucí nějakým způsobem zvýrazňovat poměrně nedůležitou informaci z rozhraní autopilota.

Výsledek návrhu

Cílem návrhu bylo realizovat takové uživatelské rozhraní, které by co nejvíce zjednodušovalo interakci pilota s aplikací. Rozhraní bylo vyvíjeno pro malá sportovní letadla, která nepotřebuji podporovat velkou množinu funkcí, jenž je přítomná u autopilotů velkých dopravních letadel. Z tohoto důvodu rozhraní obsahuje pouze základní funkční prvky autopilota. Díky jednoduchému a minimalistickému designu aplikace nepředstavuje její používání ve fázi seznámení uživatele s aplikací ani ve fázi jejího používání velké problémy.

Pro nastavování hodnoty je využit posuvník, který má pevně nastavený krok. Jako doplnění posuvníku jsou na obrazovce přítomná tlačítka pro upřesnění hodnoty. Význam těchto tlačítek je ve zrychlení procesu nastavování přesné hodnoty. Pouze s pomocí posuvníku na dotykovém displeji

není možné rychle nastavit úplně přesnou hodnotu. Proto pilot nejdříve nastaví posunem přibližnou hodnotu, kterou následně s použitím tlačítek + a − upřesní.

Aplikace byla vyvíjena pro dotykové displeje, proto bylo nutné zavést opatření proti špatnému (například náhodnému) zadávání hodnoty. Implementován tak byl mechanismus potvrzování nastavené hodnoty. K potvrzování je určeno tlačítko `CONF`, po jeho stisknutí dojde k potvrzení a zapamatování nastavené hodnoty. Pokud následně potřebujeme tuto hodnotu změnit, je nutné nejdříve pomocí posuvníku a tlačítek zvolit novou hodnotu a v časovém intervalu tuto hodnotu potvrdit. Teprve po potvrzení začne systém autopilota pracovat s nově nastavenou hodnotou. Pokud není hodnota potvrzena v určeném časovém limitu, dojde k návratu původní, naposledy zapamatované hodnoty.

Obrázek 6.7: Výsledek návrhu grafického rozhraní aplikace

Kapitola 7

Implementace

7.1 Použité nástroje

7.1.1 Programovací jazyk C++

Program byl implementován v objektově orientovaném jazyce C++. V současné době se jedná o jeden z nejrozšířenějších programovacích jazyků. Důvodem pro volbu tohoto jazyku byla právě jeho velká rozšířenost, dostupnost kvalitních knihoven pro tvorbu grafického rozhraní aplikace a jeho vhodnost pro podobný typ úloh.

7.1.2 Knihovna Qt

Knihovna Qt je multiplatformní framework určený pro vývoj aplikací s grafickým uživatelským rozhraní. Programovací jazyk C++ tento framework nativně používá, je možné jej ale použít i v dalších programovacích jazycích jako jsou C#, Java, Perl, Python. Knihovna nabízí velké množství standardních ovládacích prvků a komponent pro grafický design rozhraní. Poskytuje možnosti programování interakcí programu s uživatelem. Hlavní charakteristickou vlastnosti Qt knihovny je používání signálů a slotů pro komunikaci mezi objekty. Slot se vyvolává jako reakce na signál, který je generován změnou stavu objektu. Jedním slotem lze spojit několik signálů, a stejně tak pro jeden signál lze definovat několik různých slotů. Sloty lze použít i jako obyčejnou metodu objektu a také pro vyvolávaní signálů [21].

7.1.3 Qt Style Sheets

Pro vytváření vizuálního stylu prvků programu je možné v knihovně Qt používát Qt Style Sheets (QSS). Tyto styly jsou syntaxí i svou filosofií podobné kaskádovým stylům CSS používaným pro webdesign. QSS umožňují rychle, jednotně a jednoduše nastavovat grafický vzhled všech ovládacích prvků aplikace. Stejně jako v CSS lze měnit tvar objektu, barvu, průhlednost, vizuální reakci na událost

(např. přejetí myší nad tlačítkem) a další grafické parametry objektů. Objekty, ke kterým chceme tímto způsobem definovat styl, lze vybrat pomocí speciálních selektorů. Metodou `::setStyleSheet()` lze připojit vizuální styl k libovolnému objektu nebo k celé aplikaci [22].

7.2 Komunikační protokoly

Pro komunikaci se simulátorem jsou použity dva různé komunikační protokoly: CANaerospace a AW-COM. Přepínání mezi těmito protokoly v programu lze provádět pomocí vstupních parametrů aplikace.

7.2.1 Protokol CANaerospace

Protokol CANaerospace je vícevrstvý model založený na sériovém komunikačním protokolu Controller Area Network (CAN), který zajišťuje vysokou spolehlivost komunikace a zabezpečení, diky kterému lze tento protokol použit ve vestavěných leteckých systémech.

CANaerospace je nadstavbou nižšího protokolu CAN, a proto definuje další vrstvy modelu ISO/OSI a takovým způsobem umožňuje adresováni uzlu, potřebného pro stanovení komunikace Peer to Peer. Protokol je založen na architektuře klient-server [23].

Typy zpráv

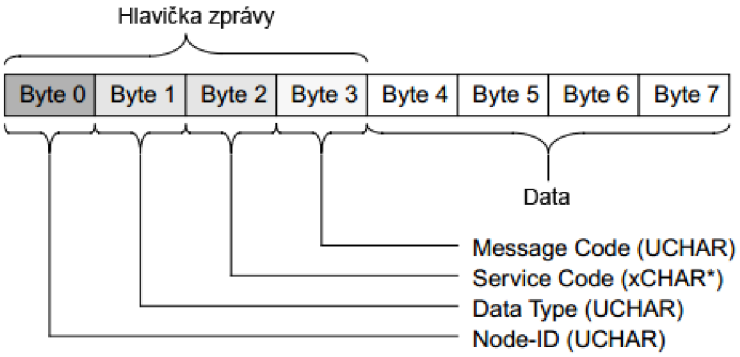
CANaerospace definuje 6 základních typů zpráv podle důležitosti. Každý druh zpráv se používá pro různé druhy služeb. Pomoci těchto zpráv probíhá komunikace mezi účastníky. Každý typ zprávy je spojen s identifikátorem CAN-ID, který definuje prioritu zprávy.

Struktura zpráv

Každá zpráva obsahuje 4 bytovou hlavičku obsahující identifikaci zprávy. Dále obsahuje 1 až 4 bytů dat. Pro prezentaci dat jsou definovány běžně používané datové typy. Byty ve zprávách jsou uspořádány v kódování Big Endian.

Formát zprávy

1. **Node-ID** – má rozsah 0-255. Definuje stanici, pro kterou je zpráva určena. Node-ID 0 znamená, že zpráva je určena pro všechny stanice,
2. **Data type** – specifikuje datový typ přenášených dat,
3. **Service Code** – slouží pro speciální data,
4. **Message Code** – číslo zprávy, slouží pro monitorování vstupních sekvencí dat.



Obrázek 7.1: Formát zprávy protokolu CANaerospace [23]

Implementace protokolu

Pro implementaci komunikace pomocí CANaerospace protokolu byla použita volně dostupná C++ knihovna libcanaero [1][24]. Knihovna poskytuje jednoduchý a rychlý přístup k protokolu na síti. Pro komunikaci se používá UDP multicast komunikaci. Pomocí této knihovny je možné rozhraní autopilota propojit s nejrozšířenějšími simulátory – Microsoft Flight Simulator a X-Plane.

Ukázka použití knihovny CANaerospace

Použití knihovny začíná vytvořením a inicializací nového objektu `SCS::Receiver` – hlavní třída pro vytváření CANaerospace komunikací, zpracovává a řídí síťové připojení protokolu. Třída `Receiver` spravuje seznam dostupných CAN datových proměnných (ukázka 7.1).

```
SCS::Scscom::Scscom()
{
   //inicializace SCS::Receiver
   const unsigned char required_sw_revision = 2;
   my_receiver = new SCS::Receiver(required_sw_revision);

   //registrace pozadavku rychlost
   my_receiver->requestData(SCS::AP_SPD_M_S::Id(), &apspeed);

   //registrace pozadavku vyska
   my_receiver->requestData(SCS::AP_ALT_M::Id(), &apalt);

   //registrace pozadavku vertikalni rychlost
```

[1] Knihovna libcanaero je volně dostupná ke stažení na adrese http://cross-simulator.com/download.php

```
    my_receiver->requestData(SCS::AP_VS_M_S::Id(), &apvs);

    //registrace pozadavku kurz HDG
    my_receiver->requestData(SCS::AP_HDG_DEG::Id(), &aphdg);

    //registrace pozadavku pozice kridelek
    my_receiver->requestData(SCS::FLAPS_POS_DEG::Id(),
    boost::bind(&SCS::Scscom::handle_flaps,this,_1), &flaps);

    //registrace pozadavku pozice vyskat
    my_receiver->requestData(SCS::TRUE_ALT_M::Id(),
    boost::bind(&SCS::Scscom::handle_althold,this,_1), &althold);

    //registrace pozadavku rezim autopilota
    my_receiver->requestData(SCS::AP_MODE_BITFIELD::Id(), &apon);
}
```

Zdrojový kód 7.1: Ukázka inicializace SCS::Receiver

Na tomto objektu můžeme volat dvě základní funkce `requestData()` a `sendData()`. Pomocí příkazu requestData propojujeme požadovaná data v třídě CAN s proměnnými v našem programu určenými pro jejich ukládání - probíhá registrace odběru dat. Pokud již máme zaregistrované všechny datové proměnné, které chceme získávat, můžeme začít volat metodu `run()`, která provádí aktualizaci dat v našich proměnných. Tuto metodu je nejlepší volat v nekonečném cyklu (ukázka 7.2), který běží jako nezávislý proces a v zadaném intervalu provádí aktualizaci dat. Aktualizaci dat můžeme také sledovat pomocí handle, která nás upozorní na změnu dat a provede příslušnou obsluhu.

```
void running::run()
{
    while(1){
        can->my_receiver->run();
        Sleep(1000);
    }
}
```

Zdrojový kód 7.2: Získávání dat

Pokud chceme naopak data odeslat z programu do simulátoru, můžeme použít metodu `sendData()`. Touto metodou nastavíme hodnotu příslušné proměnné v simulátoru a potvrdíme její odeslání metodou `run()`. Krátká ukázka kódu 7.3.

```
void SCS::Scscom::SendIAS(int IAS){
   my_receiver->run();
   apspeed = IAS;
   my_receiver->sendData(AP_SPD_M_S::Id());
}
```

Zdrojový kód 7.3: Ukázka odeslání dat

7.2.2 Protokol AW-COM

Protokol AW-COM byl speciálně vytvořen pro účely komunikace mezi softwarovými aplikacemi a simulátorem v laboratoři SimStar. Protokol je založen na architektuře klient-server a používá komunikaci pomocí TCP/IP. Komunikace probíhá pomocí zpráv [25].

Formát zpráv

Komunikační protokol AW-COM používá dva základní typy zpráv - zpráva SET a zpráva GET. Zpráva SET slouží pro odeslání dat na server, zpráva SET pro získání dat ze serveru. Obě zprávy obsahují hlavičku, data a ukončovací sekvenci. Datová část se dále dělí na část obsahující identifikace veličiny a část obsahující hodnotu. Formát zpráv 7.1.

Hlavička zprávy	Data			Ukončovací sekvence
	Hlavička	Obsah	Ukončovací sekvence	
SET	DATA	:veličina:hodnota	:END	:END
GET	DATA	:veličina:hodnota	:END	:END

Tabulka 7.1: Formát zpráv AW-COM protokolu [25]

Ukázka použití protokolu AW-COM

Spojení se serverem inicializujeme pomocí příkazu `init()`. Parametry spojení, server a port nastavíme pomocí funkce `setServer()` a `setPort()`. Ukázka inicializace 7.4.

```
Aerocom::Aerocom(int port, string server)
{
   hMutex = CreateMutex( NULL, FALSE, NULL );
```

```
    netcom.init();
    netcom.setServer(server);
    netcom.setPort(port);
    netcom.setMutex(hMutex);
    simrun = 2;
}
```

Zdrojový kód 7.4: Ukázka inicializace AW-COM

Pro odesílání dat musíme nejdříve vytvořit zprávu funkcí `setQuery()`. Následně příkazem `SetData()` odešleme zprávu na server (ukázka kódu – odeslání hodnoty autopilota (výšky) 7.5).

```
void Aerocom::SendALT(int ALT){
    //nastavime nazev veliciny jenz chceme odeslat
    string text="SET:APALT:";

    //pridame hodnotu
    text.append(netcom.DToStr(ALT));

    //ukoncime format zpravy
    text.append(":END:\n");

    //provedeme odeslani
    netcom.setQuery(text);
    netcom.SetData(&simrun);
}
```

Zdrojový kód 7.5: Ukázka odeslání dat

Pro získávání dat ze serveru slouží funkce `GetData()`, která také požaduje nejdříve sestavení požadavku a odeslání zprávy s požadavkem na server. Funkce `GetData()` vrací data do datové struktury `fsx`, ze které potom hodnotu můžeme dále používat (ukázka kódu – přijetí aktuální výšky 7.6).

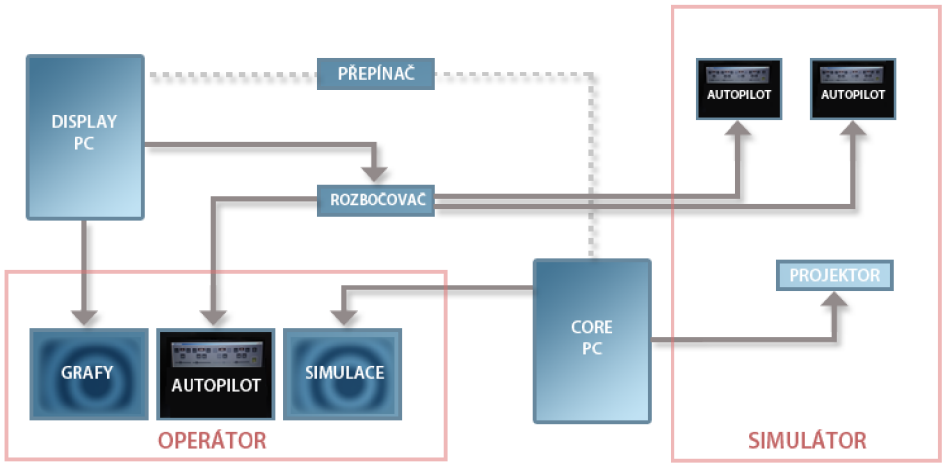
```
int Aerocom::getAltHold(){
    //nastavime nazev promenne kterou chceme ziskat
    string text="GET:AP:END:\n";
    netcom.setQuery(text);

    //ziskame a vratime hodnotu vysky
    netcom.GetData(&simrun);
```

```
    return nsDataInputInfo::fsx.altitude;
}
```

Zdrojový kód 7.6: Ukázka přijímání dat

7.3 Nasazení systémů

Na implementovaném systému bylo provedeno důkladné ověření správného fungování a bezproblémové obsluhy. Výsledný systém byl následně otestován spolu s nejrozšířenějšími leteckými simulátory – Microsoft Flight Simulator a X-Plane. Testování probíhalo také s protokolem AW-COM.



Obrázek 7.2: Diagram nasazení systému v laboratoři SimStar.

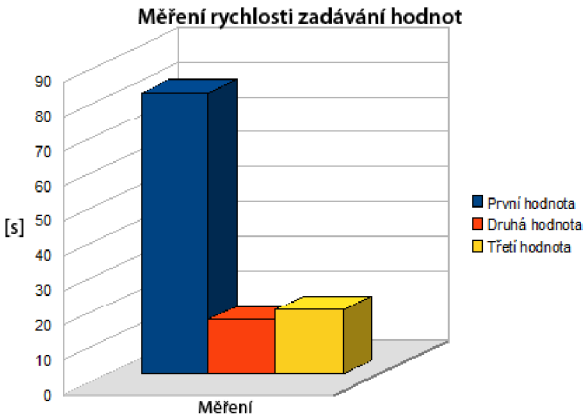
Po důkladném otestování a ladění byl systém nasazen do letecké simulační laboratoře SimStar. Diagram nasazeného systému je možné vidět na obrázku 7.2. Systém autopilota je obsluhován ze simulační kabiny letounu pomocí dotykového displeje. Zobrazen je také v místnosti operátora.

Kapitola 8

Testování

Testovaní aplikace bylo provedeno v laboratoři SimStar na leteckém simulátoru X-Plane. Prostředí laboratoře umožňovalo ověřit funkčnost programu a případně provést rychlou změnu zdrojových kódů.

Dále byla navržena sada testů pro otestovaní a vyhodnocení grafického uživatelského rozhraní. S testery byl nejdříve vyplněn stručný dotazník, obsahující otázky na jejich dosavadní zkušenosti s leteckými simulátory, autopiloty a letadly obecně. Testeři museli následně vykonat několik jednoduchých úkolů – nastavit hodnotu výšky, rychlosti a kurzu. Pro každou hodnotu byla zaznamenána doba, za kterou bylo dosáhnuto cíle. Po skončení této části ještě každý testovaný člověk ohodnotil pohodlnost, intuitivitu a grafický vzhled aplikace.



Obrázek 8.1: Průměrná doba potřebné pro zvládnutí úkolu

Testování ukázalo, že rozhraní umožňuje snadné naučení – uživatel se rychle učí používat program. Nastavení první hodnoty trvalo průměrně 80 vteřin. Při nastavování další hodnoty se ukázala vysoká efektivita naučení ovládání aplikace – doba nastavování pro dosažení cíle výrazně poklesla na průměrných 15 vteřin. Při nastavování poslední hodnoty bylo již možné pozorovat sžití uživatele se systémem – testovaní neměli již žádné problémy při nastavování poslední hodnoty, průměrně jim trvalo ji nastavit 20 vteřin. Graf průměrné doby potřebné pro provedení jednotlivých úkonů je zobrazen na obrázku 8.1

Během testování se však ukázali i některé problémy vyplývající z použitého systému ovládání pomocí dotykové obrazovky. Systém je velmi citlivý na přesnou kalibraci – je nutné nejlépe před

Obrázek 8.2: Fotografie z testování systému

každým letem provést kalibraci dotykového displeje, tak aby byli akce prováděné pilotem přesné a rychlé.

Potvrdili se také obavy z nepřesného nastavování hodnoty pomocí posuvníku. Již v návrhu aplikace bylo předpokládáno, že posuvník bude využíván pouze pro orientační nastavení hodnoty. Během testování se ukázalo, že práce s posuvníkem neprobíhá úplně podle představ pilota – reakce posuvníku nebyly zejména při plynulém pohybu s posuvníkem úplně předvídatelné.

Poslední část testování obsahovala hodnocení testera – hodnoceny byli tři kategorie pohodlnost, intuitivita a vzhled rozhraní. Každý testovaný měl možnost udělit bodové hodnocení od 0 do 10 bodů a své hodnocení doplnit slovním komentářem. Pokud vezmeme hodnocení v souvislosti s dosavadními zkušenostmi uživatelů s leteckými systémy, tak se ukázalo že uživatelé, kteří mají již nějaké zkušenosti s autopiloty byli s rozhraním velmi spokojeni. Uživatelé bez zkušeností s autopiloty byli spokojeni s pohodlností a intuitivitou ovládání, méně spokojeni byli se vzhledem aplikace.

Kapitola 9

Závěr

Cílem této bakalářské práce bylo navrhnout a implementovat grafické uživatelské rozhraní pro systém autopilota. Specifikace nového systému byla vytvářena na základě nedostatků stávajících systémů a také ve spolupráci s týmem AeroWorks, který se oblastí leteckých simulací dlouhodobě zabývá. Z této specifikace bylo dále vycházeno v části designu interakce a implementace systému.

Výsledný systém pro rozhraní autopilota splňuje všechny požadavky stanovené ve specifikaci. Systém navíc oproti specifikaci umožňuje kromě protokolu AW-COM funkčnost ještě s dalším používaným leteckým protokolem CANaerospace, který je používán jako hlavní komunikační protokol v simulátorech Microsoft Flight Simulator a X-Plane.

Nově vyvinutý systém je dobře spravovatelný a lehce rozšířitelný. Případné změny či vylepšení tak nebudou příliš složité a časově náročné. V budoucnu by se tak mohla v systému provést řada úprav a vylepšení, která budou vznikat během provozu systému. Mohl by se tak například vylepšit systém zadávání hodnot o kvalitnější a lépe použitelný posuvník. Systém je také možné rozšířit o podporu dalších leteckých simulátorů.

Implementace systému je v současné době využívána v laboratoři SimStar, kde doplňuje hlavní letový displej o možnost nastavování režimů autopilota. Systém bude v této laboratoři dále provozován a rozvíjen.

Literatura

[1] KDÉR, F. *Učebnice sportovního letce*. Druhé doplněné a přepracované vydání. Praha: Naše vojsko, 1980a. ISBN 28-017-80.

[2] JÍRA, R. *Aurodynamika a mechanika letu*. 1. vyd. Praha: Naše vojsko, 1960b.

[3] OSTOSLAVSKIJ, I. V. *Aèrodinamika samoleta*. Moskva: Gosudarstvennoe izdatel'stvo oboronnoj promyšlennosti, 1957c.

[4] TURNER, G. *Gyroscopes - Everything you needed to know* [online]. 2012d. [cit. 5. 5. 2012]. Dostupné z: <http://www.gyroscopes.org/>.

[5] WIKIPEDIA. *Gyroscopes* [online]. Wikipedia, The Free Encyclopedia, 2012e. [cit. 5. 3. 2012]. Dostupné z: <http://en.wikipedia.org/wiki/Gyroscope>.

[6] BENNETT, S. *A History of Control Engineering 1800–1930*. Pb Reprint 1986. London: Peter Peregrinus Ltd, 1979f. ISBN 08-634-1047-2.

[7] NOF, S. Y. *Springer handbook of automation*. Berlin: Springer, 2009g. ISBN 978-3-540-78830-0.

[8] HARRIS, W. *How Autopilot Works* [online]. 2011h. [cit. 17. 4. 2012]. Dostupné z: <http://travel.howstuffworks.com/autopilot.htm>.

[9] GRANT, R. G. *Flight: 100 years of aviation*. 1. vyd. London: Dorling Kindersley, 2002i. ISBN 0-7513-37323.

[10] POSID, M. C. Autopilot Systems – An Investigation of the C4I Methodologies Used in Autopilot Systems. *George Mason*. 2008j.

[11] M.A. ČERNYJ, V. K. *Samoletovoždenie*. Moskva: Transport, 1973k.

[12] Û.N. SARAJSKIJ, I. *Aèronavigaciâ*. Sankt-Petěrburg: SPbGUGA, 2010l.

[13] METCALF, T. R. *The Magnetic Sun: What is a magnetic field* [online]. 2012m. [cit. 5. 3. 2012]. Dostupné z: <http://solar.physics.montana.edu/ypop/Spotlight/Magnetic/>.

[14] MAMAEV, V. *Vozdušnaâ navigaciâ i elementy samoletovoždeniâ*. Sankt-Petěrburg: SPbGUAP, 2002n. ISBN 5-8088-077-3.

[15] VILÂEVSKAÂ, T. *Aviacionnye pribory i avtopiloty (kratkij kurs)*. Moskva: Gosudarstvennoe izdatel'stvo oboronnoj promyšlennosti, 1954o.

[16] ASAF DEGANI, M. H. Pilot - autopilot interaction a formal perspective. *Eighth International Conference on Human-Computer Interaction in Aeronautics. Toulouse, France*. 2000p, s. 11.

[17] HUGH P. BERGERON, D. A. H. Aircraft automation: The problem of the pilot interface. *NASA*. 1985q, s. 144.

[18] AEROWORKS. *AeroWorks* [online]. 2012r. [cit. 5. 5. 2012]. Dostupné z: <http://merlin.fit.vutbr.cz/AeroWorks/>.

[19] PETER CHUDÝ, K. R. Intuitive flight display for light aircraft. *AIAA Modeling and Simulation Technologies Conference, Portland, Oregon*. 2011s, s. 10.

[20] SAFFER, D. *Designing for interaction*. Second edition. Berkeley: New Riders, 2010t. ISBN 0-321-64339-9.

[21] NOKIA. *Qt Cross-platform application and UI framework* [online]. [cit. 5. 5. 2012]. Dostupné z: <http://qt.nokia.com/>.

[22] NOKIA. *Qt Style Sheets Documentation* [online]. [cit. 5. 5. 2012]. Dostupné z: <http://qt-project.org/doc/qt-4.8/stylesheet.html>.

[23] STOCK, M. *CANaerospace – Stock Flight Systems – Designed to fly* [online]. Stock Flight Systems, 2012w. [cit. 5. 5. 2012]. Dostupné z: <http://www.canaerospace.net/canaerospace.html>.

[24] MÜNZEL, P. *Libcanaero a C++ library to access CAN Aerospace over UDP* [online]. 2010x. [cit. 5. 5. 2012]. Dostupné z: <http://cross-simulator.com/doc/libcanaero/>.

[25] RYDLO, K. Vizualizace vzdušného prostoru ve 3D [online]. Diplomová práce, Masarykova univerzita, Fakulta informatiky, 2012 [cit. 2012-05-12]y. Dostupné z: <http://is.muni.cz/th/389970/fi_m/>.

Příloha A

Obsah přiloženého CD

Na přiloženém CD se nacházejí následující soubory:

- `program.zip` – zdrojové kódy a spustitelná verze programu
- `dokumentace.pdf` – dokumentace programu
- `bakalarska-prace.pdf` – text bakalářské práce ve formátu PDF
- `zdroj-bc.zip` – zdrojový tvar textu bakalářské práce