VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

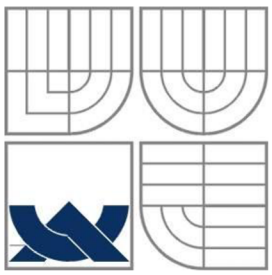
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AEROSPACE – VIZUÁLNÍ SYSTÉM INDENTIFIKACE PORUCHOVÝCH STAVŮ SYSTÉMŮ LETOUNU

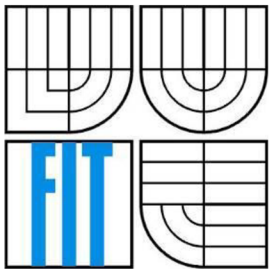
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE Stanislav Kadlčík
AUTHOR

BRNO 2010



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

AEROSPACE – VIZUÁLNÍ SYSTÉM INDENTIFIKACE PORUCHOVÝCH STAVŮ SYSTÉMŮ LETOUNU

AEROSPACE – VISUAL IDENTIFICATION OF AIRCRAFT SYSTEM FAILURE MODES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
Stanislav Kadlčík

VEDOUCÍ PRÁCE
SUPERVISOR
Ing., Peter Chudý, Ph.D. MBA

BRNO 2010

Zadání!!!!

Abstrakt

V této bakalářské práci se zabývám rozpoznáváním poruchových stavů letounu pomocí kamery. Smyslem práce je zvýšit bezpečnost letového provozu přidáním dalších bezpečnostních prvků zajišťujících informování pilota o neobvyklém stavu. Nejprve popisuji samotné rozpoznávání objektů a volbu vhodného označkování letounu pro spolehlivou detekci. V dalších částech rozvádím implementaci rozpoznávání v knihově OpenCV a popisuji svou knihovnu AirplaneRecognization.

Abstract

In this bachelor's thesis I deal with recognition of failure state of aircraft through camera recorder. The point of this thesis is to increase safety of air traffic by adding some extra safety elements which are designed to inform the pilot about abnormal states. First I describe object detection and choice of suitable markers for reliable detection. In the next parts I add details about implementation of recognition in OpenCV library and describe my own AirplaneRecognition library.

Klíčová slova

Počítačové vidění, stav letounu, rozpoznávání, detekce úhlu

Keywords

Computer vision, state of aircraft, recognition, detection of angle

Citace

Kadlčík Stanislav: Aerospace – vizuální systém identifikace poruchových stavů systémů letounu, bakalářská práce, Brno, FIT VUT v Brně, 2010

AEROSPACE – VIZUÁLNÍ SYSTÉM INDENTIFIKACE PORUCHOVÝCH STAVŮ SYSTÉMŮ LETOUNU

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Petera Chudého, Ph.D. MBA.
Další informace mi poskytli Ing. Michal Španěl a Ing. Roman Juránek.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Stanislav Kadlčík
14. 5. 2010

Poděkování

Rád bych poděkoval Ing. Peteru Chudému, Ph.D. MBA za pomoc s praktickou částí diplomové práce, za zpřístupnění letiště kdykoli jsem potřeboval a za rady, které mi poskytl ohledně pohybu a konstrukce letounu. Další poděkování patří Ing. Michalu Španělovi za informace, které mi poskytl ohledně rozpoznávání v obraze.

Obsah

1 Úvod

S rozvojem leteckého průmyslu je třeba zajišťovat bezpečnost lidí či přepravovaného materiálu. Bezpečnost se posunuje směrem, kdy obsluha nemusí sledovat každou součástku stroje, ale že samo zařízení je schopno detekovat chybu nebo v lepším případě rozpoznat chybu, která by se mohla v blízké budoucnosti objevit. Např. při spouštění počítače se provede detekce a kontrola přítomného hardwaru, aniž by si ji uživatel vyžádal. Dochází sice k mírnému zpomalení startu počítače, avšak je menší pravděpodobnost, že uživatel přijde o data vlivem hardwarové poruchy. Vrcholem snahy o detekování chyb bez zásahu člověka jsou bezpilotní letouny. Stroj je složen z mnoha mechanických součástek, které mění svoji pozici či tvar – právě pro takové případy je výhodné sledovat jejich stav vizuálně pomocí kamery.

28. dubna 1988 odlétalo z Havajských ostrovů letadlo Boeing 737-200. Necelou půlhodinu před koncem letu se ozval výbuch – byla to explosivní dekomprese poté, co se odlomil velký kus krytu trupu, takže letadlo bylo části beze střechy. S takto poškozeným letounem bylo potřeba co nejrychleji přistát. Když pilotka nechala vysunout podvozek, zjistila, že svítí jen kontrolka pravého a levého podvozku, ale přední kolo je podle signalizační diody zasunuté. Pokud se podobná situace stane v běžném provozu, letadlo má přeletět letiště, aby dispečeři zjistili, zda je podvozek vysunut, či ne. S letadlem bez střechy si tohle dovolit nemohli. Protože piloti neměli na vybranou, rozhodli se přistát i bez předního kola. Tato událost dopadla šťastně, protože přední kolo vysunuté bylo, nefungovala jen kontrolní dioda. V jiných případech však špatné letové údaje mohou způsobit katastrofu.

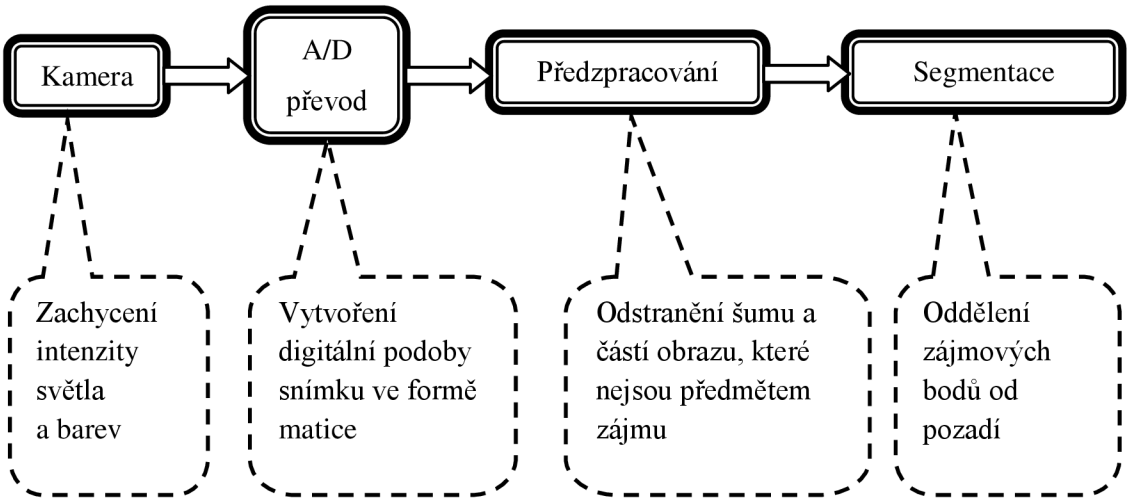
Optická detekce chyb je od začátku myšlena jako bezpečnostní prvek, který má doplňovat elektromechanické součástky, jež jsou náchylné k chybám. Výstupem kamer je proud čísel v reálném čase. Tato data jsou zpracovávána palubním počítačem. Největší výhodou vizuálního zpracovávání je, že kamera je schopna detekovat, že na zadaný cíl nevidí. Takže výsledky jsou buď správné, nebo vůbec k dispozici nejsou. U ostatních čidel je velmi obtížné zjistit, jestli naměřila skutečné hodnoty či jsou zkreslené např. námrazou ledu. Ve vizuálním zpracování se tento problém může vyřešit tak, že kamera při vyhodnocení obrazu musí detekovat známý stabilní objekt nebo značku. Pokud tento obrazec vidí, je velmi velká pravděpodobnost určení správných hodnot i u proměnlivých objektů.

Jsem přesvědčen, že v budoucnu se automatická vizuální kontrola poruch pomocí kamery a počítače rozšíří do více odvětví průmyslu.

2 Teoretický rozbor

V této kapitole je prováděn teoretický rozbor problému detekce stavu letounu. Jsou zde popsány základní vlastnosti pohybu letounu. Také jsou rozebrány jednotlivé kroky detekce, které vedou k cíly. Mezi to patří volba hardwaru pro snímání, předzpracování obrazu i samotné algoritmy pro detekci.

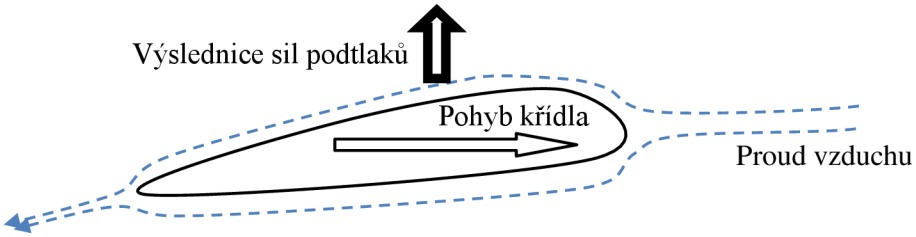
Následující graf uvádí poslounost kroků, které vedou k rozpoznání značky:



2.1 Základní údaje o pohybu letounu

Náběžná hrana křídla rozdělí proud vzduchu na část, která poletí pod křídlem, a na část nad křídlem. Horní profil křídla má vypouklejší tvar než spodní profil. Kvůli této nesymetričnosti musí částice letící nad křídlem urazit za stejný čas větší vzdálenost než ty pod křídlem. Jejich rychlost je tedy větší. Vzniká tak dynamický vztlak. Jedná se o tzv. hydrodynamický paradox, kdy v místě vyšší rychlosti vzduchu je menší tlak. Čím je rychlost obtékání plynu vyšší, tím menší je tlak v obtékaném okolí překážky.

Pod spodní částí křídla vzniká v důsledku proudu plynu nižší tlak, ale jeho síla působící na křídlo je menší než v případě podtlaku, který vznikl nad horní částí křídla. Výslednice těchto sil proto míří „nahoru“ a letoun tak může vzlétnout.



Pokud zvětšíme prohnutí profilu nebo zvětšíme horní plochu křídla, zvětší se vztlak. K tomu slouží **klapky**. Jsou to plochy umístěné v zadní části křídla blíže k trupu. Vysunutím klapky se

zvětšuje zakřivení křídla. V přímém letu (letadlo nestoupá ani nezatáčí) je nutné, aby síly, které působí proti sobě, byly vyrovnané. Suma vztlakových sil musí být rovna sumě sil působících na letadlo opačným směrem (gravitační síly…). Vztlak působící na letadlo je ovlivněn úhlem natočení klapky, ale také velikostí úhlu natočení křídla vůči směru letu a také samotnou rychlostí letu, křidélky a celkovou aerodynamikou. Samotné klapky se používají pouze při startu a přistání, kdy je potřeba co nejvíce zvýšit vztlak při relativně malých rychlostech.

Dalším prvkem pro ovládání letadla je **směrové kormidlo** (směrovka), které je umístěno na ocasní ploše. Umožňuje zatáčení letounu kolem svislé osy. K nastavení ideální pozice letadla pro letadla zatáčkou slouží **křidélka**. Je to plocha na zadní části křídla umístěná na konci (od trupu). Křidélka způsobují rotaci kolem podélné osy. V součinnosti se směrovkou však letadlo může zatáčet v ideální dráze.

Pro naklopení letadla kolem příčné osy slouží **výškové kormidlo** (výškovka) – vodorovná ocasní plocha. Výškovým kormidlem pilot reguluje výšku letu.

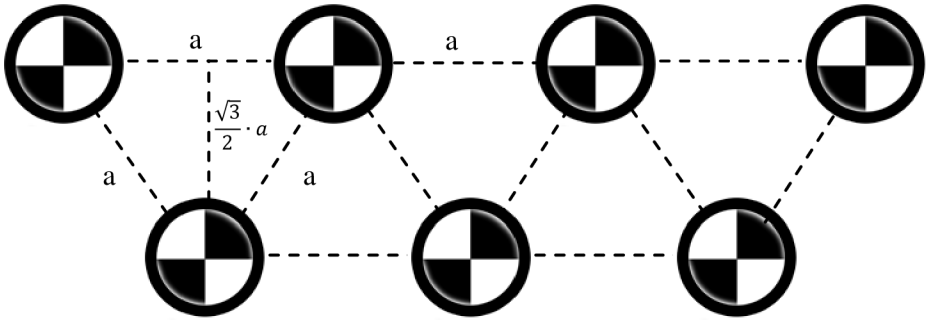
Některé typy letounů jsou vybaveny **aerodynamickou brzdou**, která zvyšuje odpor vzduchu a tím zpomaluje let.

2.2 Rozpoznání stavu

Na letoun je potřeba umístit značky pro přesné vyhodnocení zpracovávaného obrazu. Pokud by nebylo možné použít značky, celý problém by se značně zkomplikoval. Počítač by pro rozpoznání musel obsahovat 3D model letadla. Obraz z kamery by bylo třeba porovnávat s přesným modelem. Rozpoznání stavu by se velmi zkomplikovalo, protože v nejhorším případě by musel být prohledán celý stavový prostor stavů modelu a jeho korespondence s reálným obrazem.

2.2.1 Tvar značky a rozmístění

Pro určení pozice pohyblivého zařízení je dobré znát přesnou reprezentaci značek a vzdálenosti mezi nimi. Pro získání rotační matice iterativním přibližováním je přesná znalost snímaného objektu (značek) nezbytná.



Obr. 1: Rozmístění značek

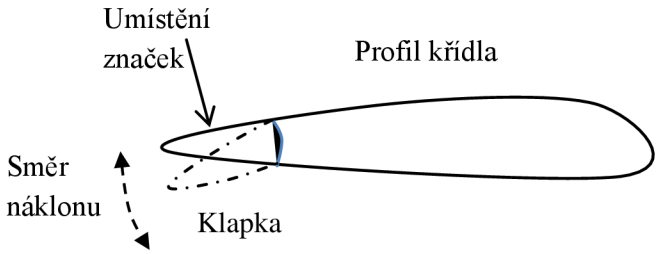
Vzdálenost mezi značkami je konstantní. Tento tvar byl zvolen tak, aby výsledný rozpoznaný bod byl co nejpřesnější. V počítačovém vidění lze jednoduše detekovat elipsu, proto kruhový tvar. Výplň kružnice je taková, aby šel s největší přesností získat střed. Detektor hran v takto zvoleném tvaru generuje 4 úsečky, které jsou všechny zakončeny ve středu značky.

2.2.2 Snímání stavu klapky

Při kalibraci značek pro daný typ křídla je v počítači uložen model obsahující body a jejich vzdálenosti mezi nimi. Tyto body korespondují se středy jednotlivých značek. Kamera (v tomto případě instalovaná v zadní části letounu) snímá značky jako elipsy, s tím, že vzdálenost mezi řadami je menší než vzdálenost mezi značkami v řadě. Pokud se klapka naklápí (směrem dolů), elipsám se zvětšuje vedlejší poloosa a vzdálenost mezi řadami se zvětšují. Toho lze využít při počítání natočení klapky – poměr mezi vzdáleností značek v řadě a vzdáleností mezi řadami určuje následující vztah:

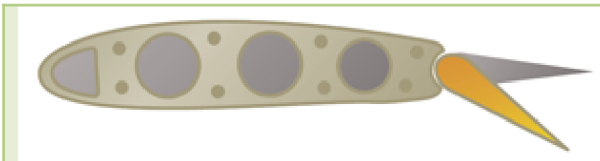
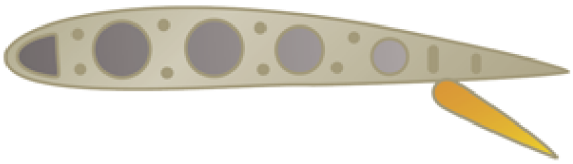
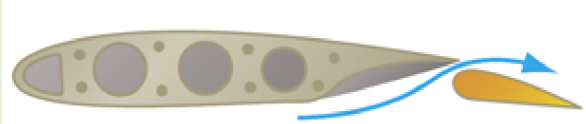
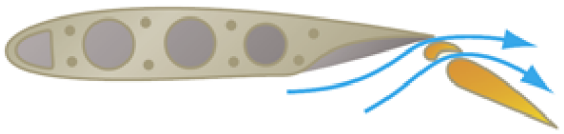
$$x = \frac{\sqrt{3}}{2} \cdot a,$$

kde a je vzdálenost mezi značkami a x je vzdálenost mezi řadami.



Obr. 2: Rotace klapky bez posunu

Seznam typů umístění a pohybů klapky:

	
Základní klapka	Oddělená klapka
	
Štěrbinová klapka	Štěrbinová Fowlerova klapka

[1]

Z obrázku je patrné, že nejsnazší snímání stavu klapky je u základní klapky, protože není třeba uvažovat vektor posunutí (klapka se nepřibližuje ke kameře). Oddělenou klapku výše uvedeným způsobem nelze snímat, protože je překryta zbylou částí křídla. Štěrbinová klapka je velmi často používána v praxi. Při vyklápění Fowlerovy klapky dochází k rotaci i jejímu posunutí. Je proto třeba sledovat vektor posunutí a ten uvažovat ve výpočtech, protože samotným posunutím se přiblíží značky ke kameře, tím se změní úhel pohledu, aniž by došlo k rotaci. V praxi je rotační i posuvný pohyb pevně svázán.

[1] Zdroj: Flight Controls. Pilot [online]. 2009, 5, [cit. 2010-05-13]. Dostupný z WWW: <http://www.faa.gov/library/manuals/aviation/pilot_handbook/media/PHAK%20-%20Chapter%2005.pdf>.

2.2.3 Alternativní umístění značky

Původní myšlenka spočívá v umístění kamery v zádi letadla a snímání jeho stavu na základě značek nalepených na povrchu klapky či jiné pohyblivé části letounu. Existují přírodní jevy, které by mohly tomuto přístupu zabránit – např. hustá mlha. Na každé kapičce mlhy vzniká rozptyl, který výrazně snižuje kvalitu výsledného obrazu, protože do čočky kamery dopadnou jen některé paprsky a to ještě s menší intenzitou. V rozmazaném obraze je mnohem složitější detekovat hrany, protože v něm prakticky nejsou ostré přechody.

Těmto nežádoucím efektům se dá předcházet umístěním kamery dovnitř křídla. Záleží na konkrétním letadle, jak se dá umístění snímacího zařízení realizovat. Některé letouny mají uvnitř palivovou či jinou nádrž (avšak i pro ně lze použít následující přístup). V zásadě je však křídlo rozděleno na spoustu komůrek. To proto, aby byla zajištěna dostatečná pevnost křídla a také proto, aby bylo letadlo dostatečně lehké, protože prostor uvnitř komor je vyplněný vzduchem (nebo palivem).

Kamerou je třeba sledovat táhla, která klapkou pohybují. Existuje několik způsobů, jak změřit, o jakou délku se vysunulo táhlo. Např. umístit značku na pohybující se část i na statickou plochu, vůči které se táhlo pohybuje. Algoritmus by pak pouze zjistil vzdálenost mezi dvěma značkami. Takto získanou délku je možné pomocí tabulky hodnot nebo interpolací funkcí převést na stupeň natočení klapky. Tento postup je striktně vázán na konkrétní typ letadla, protože malé posunutí táhla může u jednoho způsobit velký výkyv klapky a naopak. Je proto potřeba tabulka hodnot, která by v jednom sloupci obsahovala údaj o vzdálenosti mezi značkou na táhle a značkou na statické části, v druhém sloupci by byla hodnota natočení klapky. Protože v tabulce jsou diskrétní hodnoty a měření vzdálenosti značek je spojité, je třeba přepočítávat aktuální hodnotu měření:

1. k nejbližší hodnotě v tabulce
2. dopočítat lineárně (či jinou funkcí) na základě 2 nejbližších sousedů
3. body tabulky proložit funkcí, která je spojitá

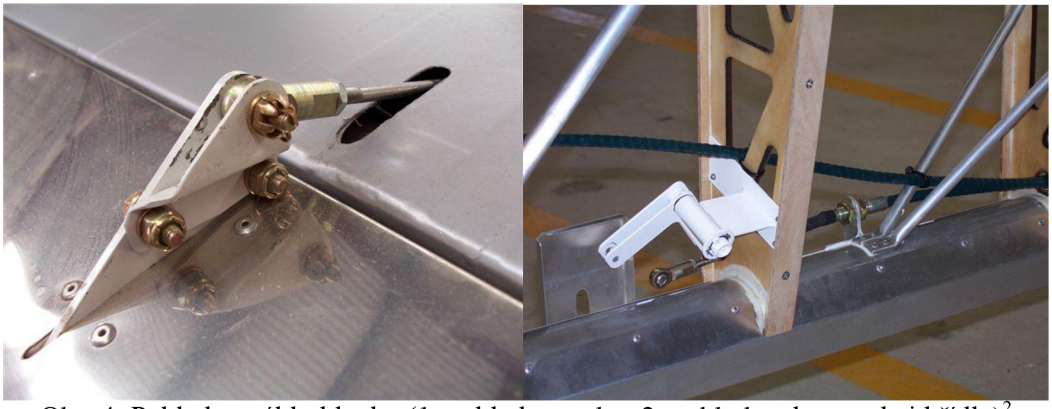
Pro bod 3 existuje několik řešení:

- Lagrangeova forma interpolačního polynomu
- Newtonův interpolační polynom
- Interpolace pomocí splajnů

Pro menší počet bodů jsou vhodné první dvě zmíněné interpolace. Prokládání body splajny je nejobecnější a nejpřesnější řešení. Protože se interpolační křivka počítá pro každý typ letounu pouze jednou, je vhodné použít co nejpřesnější metodu.

Nevýhodou umístění kamery dovnitř křídla je fakt, že je třeba zajistit zdroj osvětlení. Dalším záporem metody je složitější údržba (čištění objektivu apod.).

Obr. 3: Oblast zájmu při snímání „zevnitř“



Obr. 4: Pohled na táhlo klapky (1. pohled z venku; 2. pohled na konstrukci křídla)[2]

Obr. 5: Ovládací zařízení výškového kormidla letounu Zlín Z-242L

2.3 Hardware pro snímání obrazu

Nejdůležitější částí vizuálního systému je snímač obrazu. V zásadě se nabízí 2 možnosti. Buď použít fotoaparát, který na vyžádání palubního počítače vyfotí snímek. Jako vhodnější a jednodušší řešení se ukázalo použití kamery, která do palubního počítače posílá 25 snímků za sekundu. Tyto snímky je lépe zpracovávat nezkomprimované, protože tím snižujeme požadavky na dekódovací hardware.

2.3.1 Princip snímání kamerou – výběr technologie

Digitální kamera je zařízení, která analogový signál v podobě světla převádí na digitální signál s popisem jednotlivých bodů v matici snímku. Nejdůležitější částí kamery je senzor převádějící světlo na elektrické impulsy, se kterými již kamera umí dále pracovat.

Nejčastěji používané typy senzorů:

- CCD (Charge-Coupled Device) – tento snímač je schopný uchovávat náboj delší dobu a přitom je velmi citlivý. Využívá fotoelektrického jevu, kdy foton převede elektron do excitovaného stavu. Předaná energie elektronu závisí na vlnové délce fotonu ($E = f \,.\, h$, kde h je Planckova konstanta, f je frekvence vlnění fotonu). Excitovaný elektron způsobí vodivost

[2] Zdroj: Aeropup/Supapup [online]. 2004 [cit. 2010-05-13]. Aeropup Light Aircraft. Dostupné z WWW: <http://www.aeropup.com/pic_viewer.php>.

a elektrický proud, který zaznamenají elektrody. Aby nedocházelo k úniku excitovaných elektronů, je polovodič od elektrod oddělen dielektrikem SiO_2.

- CMOS (Complementary Metal–Oxide–Semiconductor) – pracuje na podobném principu jako CCD (rovněž obsahuje elektrodu a polovodič, na který dopadá světlo), avšak světlocitlivé buňky jsou u aktivní verze doplněny obvodem, který je schopný snižovat šum (APS - Active Pixel Sensors), protože pasivní verze generuje velké množství šumu. CMOS na rozdíl od CCD dovede pracovat s pouze jednou napěťovou úrovní, čímž klesá potřeba energie. U obou snímačů platí, že čím méně se zahřívají, tím méně šumu vzniká. Protože CMOS snímač není tak náročný na elektrický proud, nezahřívá se tolik a tím tolik nevzrůstá šum.

CCD vyniká rychlostí zpracování snímku a kvalitou obrazu, CMOS zase nízkou spotřebou energie a nízkými náklady na výrobu. Pro rozpoznání poruchových stavů je výhodnější použít CCD z důvodu méně zašuměného obrazu.

2.3.2 Základní vlastnosti kamery

Je třeba zvážit způsob snímání obrazu kamerou:

- **Progresivní** – kdy se ukládají celé snímky najednou a posílají jako celek
- **Prokládané řádkování** – snímek je v podobě každého druhého řádku. Má tedy poloviční velikost, ale je potřeba snímat s dvojnásobnou frekvencí. Při pohybu objektu dochází k rozmazání obrazu, protože jedna část už zachytila změnu pozice objektu, druhá část však čeká na překreslení.

Pro jednodušší zpracování obrazu je výhodnější použít kameru s progresivním snímáním obrazu. V opačném případě (prokládané řádkování) by bylo třeba čekat na následující půlsnímek, složit je v jeden celek, a pak teprve aplikovat rozpoznání objektů ve videu.

Dalším významným kritériem je rozlišení obrazu. Snímané značky na objektu je potřeba zachytit v dostatečné kvalitě. Volba rozlišení kamery závisí na vzdálenosti kamery od objektu a na velikosti značek, které kamera snímá.

Následující tabulka udává, jaký poměr musí být mezi velikostí značky a vzdáleností značky od kamery, aby detekce byla spolehlivá. Tabulka je uvedena pro 2 nejpoužívanější rozlišení SD (720:576px) a HDV (1440:1080px, poměr stran pixelu (Pixel aspect ratio) je 1,3).

Rozlišení + velikost značky/vzdálenost [m]	**1 m**	**3 m**	**5 m**	**7 m**	**9 m**
DV PAL 5 cm	Ano	Ne	Ne	Ne	Ne
HDV PAL 5 cm	Ano	Ano	Ne	Ne	Ne
DV PAL 10 cm	Ano	Ano	Ne	Ne	Ne
HDV PAL 10cm	Ano	Ano	Ano	Ne	Ne

2.4 Předzpracování obrazu

Obraz získaný z kamery je potřeba předzpracovat, aby rozpoznání bylo co nejpřesnější. Při snímání kamerou je asi nejčastějším problémem nízká intenzita osvětlení a naopak příliš vysoká, kdy vzniká přepálený obraz. Tato kapitola pojednává o metodách úpravy obrazu pro pozdější zpracování.

2.4.1 Globální prahování

Globální prahování je metoda pro převod typicky obrazu v odstínech šedi do varianty pouze s dvěma barvami (černá a bílá). Vychází z předpokladu, že celý obraz v odstínech šedi můžeme rozdělit do dvou skupin – na body s větší intenzitou než je zvolená číselná hodnota (práh) a s nižší intenzitou. Pokud je intenzita bodu menší než práh, je jeho barva nastavena na černou. V opačném případě na bílou.

Číselná hodnota prahu se získává z histogramu. Ideální případ pro globální prahování je ten, kdy plocha histogramu je z větší části kolem dvou bodů, které jsou ve stupnici odstínů šedi (např. 256 hodnot) dál od sebe. Za práh se v tomto případě volí hodnota, která je mezi těmito dvěma body s největší hustotou. Histogram s takovým průběhem hustoty se nazývá bimodální. Jedná se ovšem o ne příliš častou hustotu rozložení intenzity bodů fotografie reálného světa. Proto se častěji používá prahování adaptivní.

2.4.2 Adaptivní prahování

Tato metoda se snaží obejít vlastnost, že obraz reálného světa ve většině případů nejde rozdělit konkrétní hodnotou (prahem) na bílou a černou, ale pro jednotlivé části obrazu je třeba získat práh vhodný pro tu kterou část. Vstupem této metody je obraz převedený na stupně šedi a výstupem je dvojbarevný obraz (černá, bílá). Adaptivní prahování lze s výhodou použít, když některá část obrazu je ve stínu nebo když část je více osvětlena než ty zbylé. Výsledkem je několik prahových hodnot pro jeden snímek. V praxi se práh počítá pro každý bod obrazu zvlášť z intenzity daného bodu a z funkce, která popisuje jeho okolí (intenzitu světla). Funkce popisující okolí bývá nejčastěji vypočítána jako aritmetický průměr zvoleného okolí.

2.4.3 Jasová korekce

Pokud je známa analytická plocha intenzity jasu v prostoru, stačí tuto plochu odečíst od původního snímku. Jestliže toto analytické řešení není známo, je třeba získat opravné koeficienty. Ty vycházejí ze znalosti jasu určité plochy obrazu (etalonová plocha).

Multiplikativní model poruchy: nový jas $f(m,n) = e(m,n) \cdot g(i,j)$, kde g je vstupní obraz a e je jas etalonové plochy.

Opravné koeficienty se získají jako

$$koeficient(m,n) = e(m,n) \cdot c, kde\ c\ je\ známý\ jas \qquad (1)$$

2.4.4 Eliminování deformace obrazu

Při výrobě kamery je mnohem jednodušší vyrobit „kulovitou“ čočku než matematicky ideální parabolickou. Chyba bývá také způsobena nepřesnou konstrukcí objektivu či kamery, dále také špatným uložením čoček a nízkou kvalitou antireflexních vrstev. Složitější objektivy bývají složeny z deseti a více čoček (rozptylek) – vzniká tak deformace obrazu nepřesným rozmístěním. Tato nepřesnost se projevuje jako tzv. soudkovitý (barrel) obraz nebo efekt „rybího oka“ (fish-eye). Paprsky, které jsou od středu obrazu vzdálenější, jsou více zkresleny a vzdálenosti mezi jednotlivými body neodpovídají poměru k bodům uprostřed obrazu a reálnou skutečností.

Při rozpoznávání stavu letounu existují v zásadě 3 možnosti, jak tuto chybu napravit nebo alespoň snížit její následky:

- Umístění zkoumaného objetu co nejblíže středu – radiální deformace ovlivňuje hlavně okraje obrazu. Pokud by se podařil zkoumaný objekt umístit do středu snímaného (kde je deformace rovna nule), vzdálenosti mezi body nejsou zkreslené nebo jejich zkreslení je zanedbatelné. Daný postup však zanáší chybu, kterou je možno ještě zmenšit použitím kamery s velmi vysokým rozlišením a přitom malou radiální deformací. Při zkoumání několika málo pixelů v rámci celé obrazovky je zkreslení malé. Pro přesný výsledek je potřeba použít jeden z následujících postupů:
- Použití objektivu s velmi přesnou parabolickou čočkou (s minimální radiální deformací) – finančně náročné
- Matematickou korekcí – v praxi je deformace uvnitř obrazu popsána několika prvními výrazy Taylorova polynomu. Koeficienty, které jsou nezbytné pro charakteristiku distorze, jsou r^2, r^4, někdy i r^6 (u velkých deformací typu „rybí oko“). Obecně lze přeskupení bodů obrazu vyjádřit těmito rovnicemi:

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \quad (2)$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6), \quad (3)$$

kde $k_{1\text{-}3}$ jsou koeficienty radiálního zkreslení; $r^2 = \overline{x}^2 + \overline{y}^2$.

$$\overline{x} = \frac{x - s_x}{\sqrt{s_x^2 + s_y^2}}, \overline{y} = \frac{y - s_y}{\sqrt{s_x^2 + s_y^2}}, \quad (4)$$

kde s_x , s_y jsou souřadnice středu

Algoritmus úpravy deformace postupně prochází bod po bodu a z výše uvedených vztahů vypočte novou souřadnici. V zásadě se jedná o reálné číslo; indexy matice obrazu jsou přirozená čísla. Proto je nutné provést interpolaci. Pro jednoduchost lze výsledek zaokrouhlit, pro přesnější výsledek je možné využít bilineární interpolaci.

2.5 Detekce zájmových bodů

Detekcí zájmových bodů se rozumí hledání objektu, který splňuje alespoň jednu z následujících podmínek:

- Je přesně známa pozice objektu v obraze
- Objekt je přesně definován (nejlépe matematicky)
- Okolí zájmového bodu má lokální obrazovou strukturu bohatou na informace o obsahu (obsahuje dostatečné množství specifických bodů)
- Je stabilní jak v částečných tak globálních poruchách (perturbacích) natolik, aby zájmový bod mohl být opakovaně spolehlivě detekován. Perturbace zahrnují změny způsobené transformacemi (perspektivní a jiné projekce, rotace, posuvy, změny velikosti atd.), různou úrovní osvětlení…

2.5.1 Detekce hran

Lidské vnímání světa zrakem je založeno na rozpoznávání hran. Na základě tohoto vjemu mozek vytvoří představu objektu, na který se právě díváme. Hrana může být v místě, kde dochází k náhlé změně jasu (ne však nutně). Stejně jako lidské vnímání pracují segmentační techniky zpracování obrazu. Detekce hran je základní částí oddělení objektů. Za objekt je považována ta část obrazu, která je zajímavá z hlediska dalšího zpracování.

Základní vlastnosti **hrany**:

- Je určena vlastnostmi změny jasu v obraze a okolím této změny
- Popisuje rychlost změny obrazové funkce *f(x, y)*.

Bod určující hranu (hranový bod), je bod s velkou změnou gradientu obrazu. Ne však každý bod, kde se intenzita světla prudce mění, je hranový bod.

Způsoby hledání hran v obraze jsou založeny na hledání extrémů obrazové funkce. Toho je možné dosáhnou buď hledáním maxima první derivace a porovnáním hodnoty s prahem (Canny) nebo položením druhé derivace rovné nule (Marr-Hildreth), která je však náchylnější na šum. Je potřeba vyhlazovat obraz tak, aby hrany byly dostatečně ostré a přitom aby byl odstraněn šum.

Gradient u jednorozměrných funkcí je derivací. Obraz je však dvojrozměrný, proto je gradientem vektor parciálních derivací (určuje velikost a směr).

$$\|\nabla f(x,y)\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}, (\sin\psi, \cos\psi) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) \tag{5}$$

U diskrétních signálů je potřeba průběh aproximovat funkcí. Její derivace už lze jednoduše spočítat. Další možností je aproximace derivace konečnými diferencemi.

Gradient (vektor velikosti a směru) může být vypočten pomocí konvolučního filtru. Filtr je v podobě matice. Její prvky udávají váhy jednotlivých bodů. Při vysoce zašuměném obraze je lepší použít filtr s větší maticí, protože zahrnuje větší okolí. Nejlepší výsledky konvoluce jsou pomocí masky, která je vytvořena pro daný šum v obraze.

Některé typy filtrů:

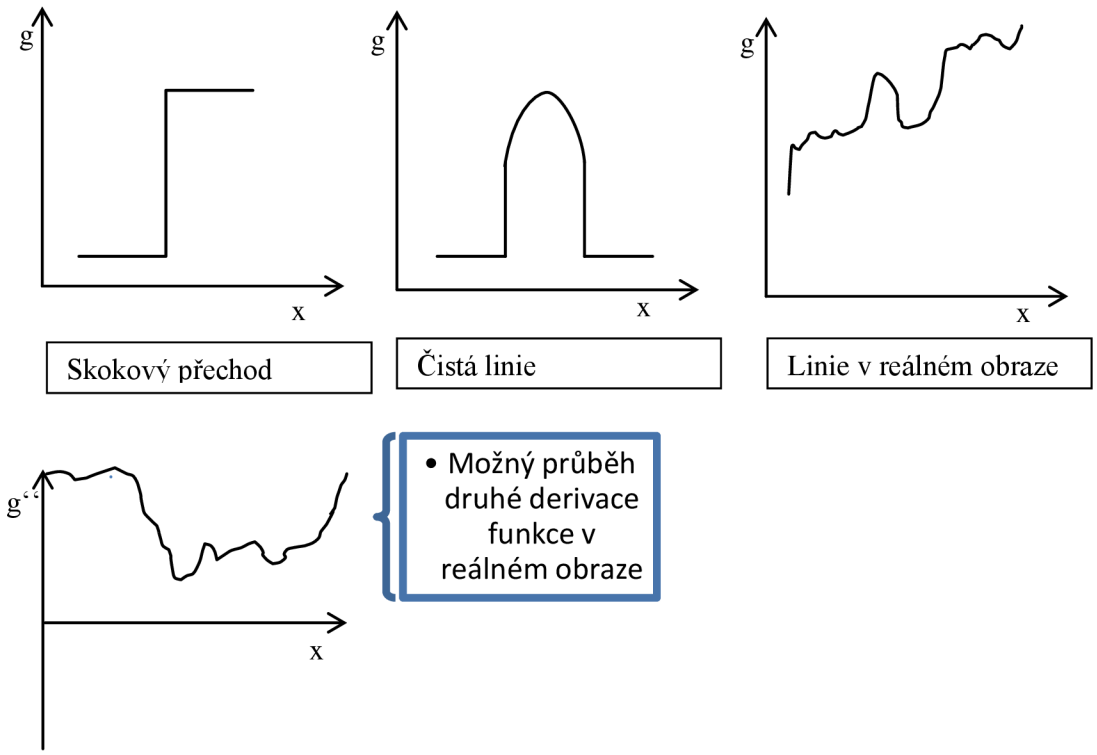
Robinsonův operátor:

$$h_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}, h_2 = \begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -2 & -1 \end{bmatrix}, h_3 = \begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix} \tag{6}$$

Kirschův operátor:

$$h_1 = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix}, h_2 = \begin{bmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{bmatrix}, h_3 = \begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix} \tag{7}$$

Existuje ještě mnoho dalších, které zde neuvádím, protože principiálně jsou stejné jako výše zmíněný příklad. Jejich odlišnost je pouze v tom, jakou váhu dávají okolním bodům.



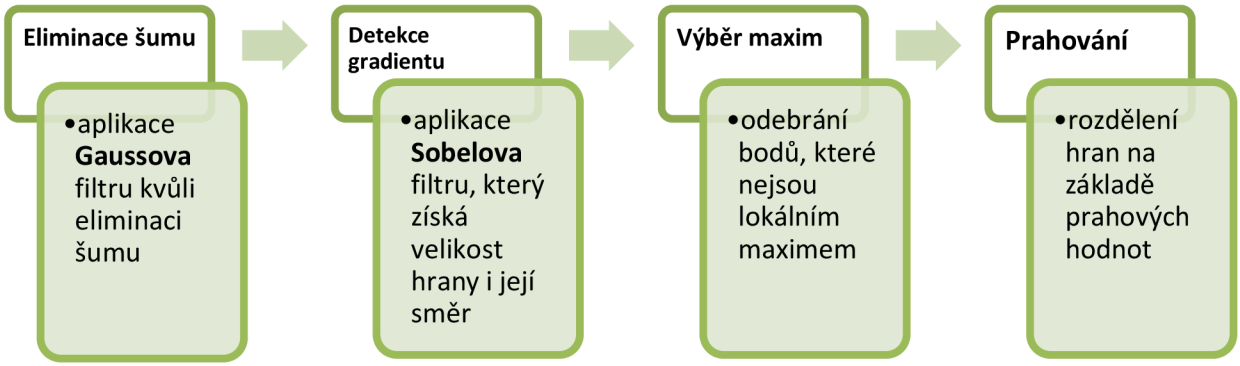
Obr. 6: Průběh funkcí v okolí hrany

Ačkoli druhá derivace funkce protíná osu x na několika místech, správně detekovaný hranový bod je pouze jeden.

2.5.2 Cannyho hranový detektor

Algoritmus, který splňuje základní podmínky detekce hran:

- Každou hranu detekuje pouze jednou (výběr maxim)
- Přesně určuje velikost i polohu hrany (detekce gradientu)
- Snižuje chybovost (prahování)



Pro **detekci gradientu** je nejvhodnější použít Sobelův filtr, protože není tolik citlivý na šum. Jeho konvoluční matice má tvar:

$$h_1 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \qquad h_3 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & -0 \end{bmatrix} \tag{8}$$

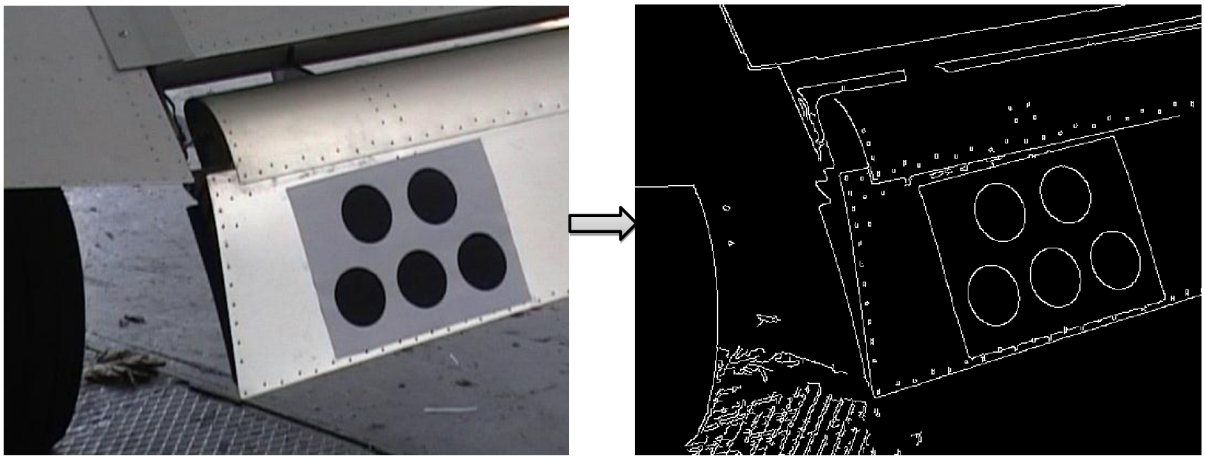
Konvoluce tímto filtrem je aplikována v jednom směru (pro h_1 vodorovně, pro h_2 svisle…).

Výběr maxim (thinning) sleduje hranové body. Okolí hranových bodů musí splňovat, že hodnota gradientu je menší než v samotném hranovém bodě (okolím se neuvažují ostatní hranové body). Pokud tato podmínka splněna není, pak je hrana zamítnuta.

Vlivem šumu jsou označeny i hrany, které hranami nejsou a při detekci je nechceme uvažovat. K rozdělení na správné hrany a nechtěně detekované hrany slouží prahování (thresholding). Hrany rozdělíme na 3 stavy:

- Správná hrana
- Hrana na pomezí – její „správnost“ závisí na okolí
- Špatně detekovaná hrana

Tyto stavy jsou rozděleny prahy P_1 a P_2, které určují intenzitu hraničních gradientů. Pokud gradient zkoumaného bodu je větší než P_2, pak je brán jako hranový. Pokud je menší než P_1, pak je bod zamítnut. Pokud se aktuální bod nachází v intervalu (P_1, P_2), pak je za hranový uznán tehdy, pokud sousedí s již uznaným hranovým bodem. Tuto část algoritmu lze jednoduše implementovat rekurzivní funkcí. Je však potřeba zajistit, aby nedocházelo k cyklickým vazbám, kdy jeden bod čeká na potvrzení souseda jako hranového bodu a ten zase čeká na původní bod.



Obr. 7: Aplikace Cannyho hranového detektoru pomocí knihovny OpenCV

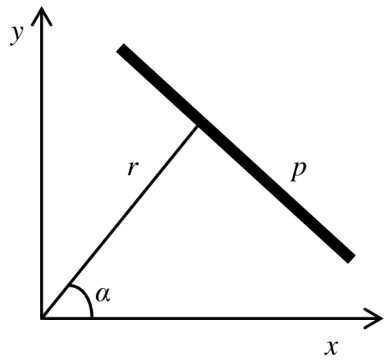
2.5.3 Detekce přímky – Houghova transformace

Houghova transformace popisuje parametrické vlastnosti objektu v obraze. Výhodou Houghovy transformace je částečná odolnost proti nepravidelnostem tvaru a jeho přerušení. Před samotnou transformací je potřeba detekovat hrany. Vstupem metody je digitální obraz s hranami, výstupem je analytické vyjádření objektů.

Mějme rovnici popisující přímku

$$x \cdot \cos\alpha + y \cdot \sin\alpha = r \tag{9}$$

Následující obrázek vysvětluje parametry rovnice:



Obr. 8: Přímka v soustavě souřadnic (*x, y)*

Mějme body $[x, y]$ obrazu, které dosaďme do analytické rovnice přímky. Pak množina všech řešení (*α, r*) Houghova prostoru vytvoří spojitou křivku. Po promítnutí všech bodů ležících na přímce *p* do Houghova prostoru všechny křivky odpovídající bodům $[x, y]$ se protnou v maximální hodnotě parametrů α_{max}, r_{max}. Protože hledáme 2 parametry, Houghův prostor je dvojdimenzionální.

2.5.4 Detekce elipsy

Při detekci Houghovou transformací je potřeba znát analytický popis. Pro elipsu je to následující rovnice:

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1, \tag{10}$$

kde $[x_0, y_0]$ jsou souřadnice středu a *a, b* jsou po řadě délky hlavní a vedlejší poloosy.

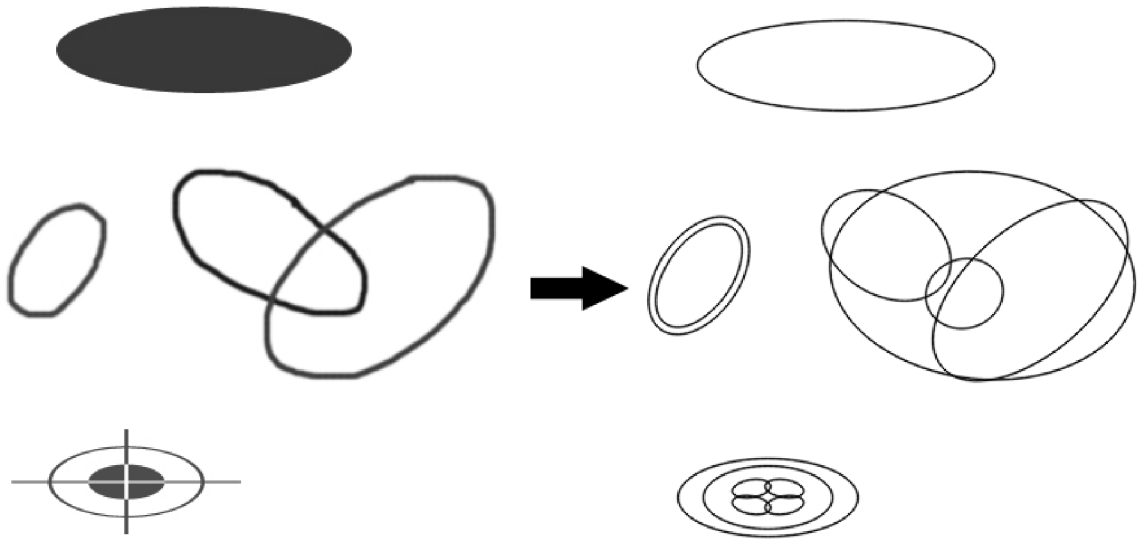
Parametrická rovnice elipsy je:

$$\begin{aligned} x &= x_0 + a\ \cos\theta \\ y &= y_0 + b\ \cos\theta \end{aligned} \tag{11}$$

kde θ určuje úhel natočení elipsy.

Způsob detekce elipsy je obdobný postupu hledání přímky. Protože však hledáme 5 parametrů, roste složitost prohledávání (časová i paměťová). Houghův prostor je dimenze pět. Po promítnutí všech bodů ležících na elipse *p* do Houghova prostoru všechny křivky odpovídající bodům $[x_i, y_i]$ se protnou v maximální hodnotě parametrů $x_{0_{max}}$, $y_{0_{max}}$, a_{max}, b_{max}, θ_{max}.

Rozměr 5 dimenze Houghova prostoru se stává kvůli časovým a paměťovým nárokům nepoužitelný. Proto se pro hledání elipsy používá **modifikovaná Houghova transformace.** Modifikace spočívá v tom, že se nejprve odstraní všechny rovné hrany. Elipsa nemůže obsahovat žádné úsečky. Hlavní změna oproti klasické Houghově transformaci je v tom, že se iterativně procházejí 2 parametrické prostory, aby se nalezly vrcholy hran – a tedy body ležící na elipse.



Obr. 9: Detekce elipsy pomocí knihovny OpenCV

2.6 Sledování změny stavu

Ke sledování značek umístěných na letadle jsou využity výše uvedené postupy. Nejdůležitější částí celého projektu je však zachytit změnu stavu. Základem pro určení pozice je znalost bázového 3D modelu. V paměti počítače musí být uložena poloha referenčních bodů, které jsou rovněž vidět v obraze nebo tabulka vzdáleností mezi body.

2.6.1 Projekce do 2D

Protože při zjišťování rotační matice se pracuje s transformací reálného světa do dvourozměrného obrazu, je potřeba zmínit definici projekce. Hlavní myšlenkou je transformace 3D objektu do 2D transformační maticí P:

$$P = \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{12}$$

Transformace mezi 2D bodem A = (a_x, a_y) a 3D bodem B = (b_x, b_y) má následující tvar:

$$\begin{pmatrix} a_x \\ a_y \\ z \end{pmatrix} = \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} b_x \\ b_y \\ b_z \\ 1 \end{pmatrix}, \tag{13}$$

kde z je faktor poměru, který je použit při normalizaci 2D bodu A (a_x/z, a_y/z). Hodnoty f_x a f_y jsou ohniskovou vzdáleností v *x* respektive *y* ose.

2.6.2 Zjištění rotační matice

Zadání problému – v prostoru se otáčející část letadla – se jeví jako obecně analyticky řešitelné. Je znám přesný model značek, kamera a detekční algoritmus zjistí polohu značek v perspektivní projekci. Problém se tady dá převést na hledání rotační matice a translačního vektoru z aktuálních naměřených hodnot. Polohy značek se na obraze mění v důsledku perspektivní projekce a proto je

možné buď vytvořit tabulku pozic a jim odpovídajícím úhlům, nebo na základě homografie obrazu spočítat parametry natočení v prostoru.

Pro zpracování změn úhlu za účelem zjištění rotační matice je potřeba vytvořit referenční 3D model značek. Existují 2 možnosti, jak jej získat:

- Přesným určením poloh značek v souřadnicovém systému prostoru
- Rekonstrukcí z perspektivní projekce zanedbáním z-souřadnice

Prvně jmenovaný způsob znamená, že se ohodnotí vzdálenosti, posunutí a rotace mezi značkami samotnými a mezi značkami a kamerou, pokud její čočku bereme jako počátek soustavy souřadnic. Vytvoří se tedy model, vůči kterému je pak referencována každá poloha.

Druhá metoda vychází z úvahy, že při výkyvu klapky letadla se minimálně mění vzdálenost mezi značkami a kamerou. Referenční trojdimenzionální model je vytvořen z 2D obrazu, který snímá kamera. Tím se získají souřadnice x a y. Za souřadnici z je zvoleno konstantní číslo. Pro vyčíslení pozice je použit iterační algoritmus. Ten bere v úvahu model i aktuální polohu značek. Iterativně přibližuje referenční model k současné poloze a skončí, jakmile jejich podobnost dosáhne zvolené hranice. Tento postup nezjišťuje rotační matici značek vůči objektivu kamery, ale vůči jiné referenční perspektivní projekci.

POSIT je algoritmus pro zjištění natočení a posunutí známého 3D objektu ve 2D obraze. Kombinuje 2 algoritmy: POS (Pose from Orthography and Scaling) a POSIT (POS with iterations). Prvně jmenovaný aproximuje perspektivní projekci ortografickou projekcí a nalézá rotační matici a vektor posunutí řešením lineárního systému. POSIT je iterační algoritmus, který používá POS. V iteraci posílá algoritmu POS jako originální objekt výsledek předchozího výpočtu a tím zpřesňuje určení pozice. POSIT obvykle provede 4-5 iterací, než dosáhne výsledku. Přesný počet průchodů závisí na přesnosti, s jakou chceme určit výslednou rotační matici. POSIT není náročný na zpracování. Obsahuje nenáročné operace v plovoucí řádové čárce a je tedy vhodný pro zpracování v reálném čase.

Výhodou algoritmu je, že jako vstup nevyžaduje počáteční odhad výsledku na rozdíl od klasických Newtonových metod.

V první iteraci POSIT hledá pozici násobením vektoru bodů modelu dvěma vektory (vektor_obrazovych_bodu.x, vektor_obrazovych_bodu.y). Výsledkem této operace jsou dva vektory, které po znormalizování jsou přímo výsledkem prvních 2 řádků rotační matice. Průměr těchto dvou vektorů je měřítko projekce, které se použije pro výpočet posunu. V další iteraci se v zásadě provedou stejné operace, avšak už s pozměněným obrazovým vektorem. Pro přesné vyjádření pozice POS algoritmus vyžaduje SOP (Scaled Orthographic Projection = pravoúhlá projekce v měřítku) namísto perspektivní projekce. Proto se ve druhé iteraci přepočítává projekce do SOP. Za obrazový vektor je brán vektor z předchozí iterace přepočítaný do pravoúhlé projekce.

Rotační matice je ve tvaru

$$R = \begin{bmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{bmatrix} \tag{14}$$

Pseudokód algoritmu POSIT. Jako vstup jsou brány `bodyObrazu`, `bodyModelu` a ohnisková vzdálenost, jako výstup pak matice `rotace` a vektor `posun`.

```
POSIT (bodyObrazu, bodyModelu, ohniskovaVzdalenost, &rotace, &posun)
{
  pocet  = 0;
  ukonci = 0;
  vektorModelu = bodyModelu - bodyModelu[0]; // bodyModelu[0] je referencni bod
  zalohaSOPBodyObrazu = bodyObrazu;
  while (!ukonci)
  {
    if (pocet == 0) // v prvni iteraci
    {
       // pro kazdy bod obrazu spocitej rozdil s referencnim bodem
       // uloz do vektoru o stejne velikosti jako je velikost bodyObrazu
       // bodyObrazu[0] je referencni bod (0, 0, 0)
       obrazovyVektor = bodyObrazu - bodyObrazu[0];
    }
    else      // pocet > 0
    {
                              // .*   = nasobeni pro kazdy prvek
      SOPBodyObrazu = bodyObrazu .* ((1 + vektorModelu * radek3/posun[3]));
      // SOPBodyObrazu[0] = referencni bod
      obrazovyVektor = SOPBodyObrazu - SOPBodyObrazu[0];
      zalohaSOPBodyObrazu = SOPBodyObrazu;                      // vytvoreni zalohy

      rozdilObrazu = sum(sum(abs(round(SOPBodyObrazu) - round(zalohaSOPBodyObrazu)))

    }
    // pocitam zvlast pro x a y souřadnici bodu
    IVektor = pseudoinverse(vektorModelu[][x]) * obrazovyVektor[][x];
    JVektor = pseudoinverse(vektorModelu[][y]) * obrazovyVektor[][y];

    // vektor norm (vektor) { return sqrt(vektor * vektor); }
    radek1 = IVektor / norm(IVektor);
    radek2 = JVektor / norm(JVektor);
    radek3 = vektorovySoucin(radek1, radek2);
    rotace = [radek1; radek2; radek3];
    pomer = (norm(IVektor) + norm(JVektor)) / 2;
    posun = [bodyObrazu[1][x]; bodyObrazu[1][y]; ohniskovaVzdalenost] / pomer;

    // jestlize nejsem v prvni iteraci a rozdil je mensi nez 1
    if ((pocet > 0) && (rozdilObrazu < 1))
      ukonci = true;
    else ukonci = false;
       pocet = pocet + 1;
  }
  return;
}
```

2.6.3 Měření úhlu ze vzdálenosti 2 bodů

Projekt měření pohybových stavů letadla je úzce specifický problém a vyžaduje přesně nastavené a kalibrované parametry. Kamera je fixně umístěna na jedné pozici; „dívá“ se pořád na stejné místo. Přesnost výpočtu lze zvýšit optickým přiblížením.

Jednou z možností, jak snímat stav klapky, je umístit jednu jedinou značku na pohyblivou část. Při kalibraci dochází k uložení snímaných dat do databáze programu. Pokud je tedy klapka natočena o nejvyšší možný úhel, perspektivní projekce značky je na souřadnicích *x*, *y* obrazu. Stejně tak můžeme postupovat pro různé úhly. V praxi se tato technika ukázala jako velmi přesná, pokud je letadlo v klidu. Za letu však dochází k deformacím křídla. Ačkoli je spousta výrobních materiálů (laminát, plech, hliníkové slitiny, uhlíkové výztuže) obecně se křídlo za letu pohybuje vůči trupu,

protože je vystaveno velkým aerodynamickým tlakům. Klapka tedy zůstane v klidu, ale protože došlo k deformaci materiálu a tedy změně polohy perspektivní projekce klapky, je detekována změna úhlu.

Uvedený postup je možné upravit tak, aby eliminoval deformace materiálu. Na klapku či jiné zkoumané zařízení se umístí 2 značky tak, aby od sebe měly co největší vzdálenost. To, co v současném stavu snímá algoritmus, není poloha značek v prostoru, ale jejich vzdálenost. V reálném 3D prostoru vzdálenost zůstává pořád stejná, ale v perspektivní projekci promítnuté kamerou do 2D obrazu se vzdálenosti mění.

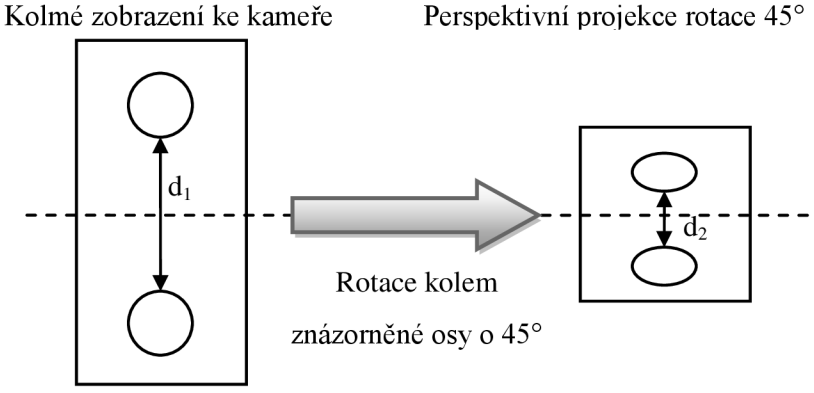
Výhodou postupu je, že nepotřebuje kalibraci kamery. Není třeba odstraňovat soudkovitost obrazu ani jiné vady. Nevýhodou je, že se nejedná o obecné řešení, které by rozpoznalo změnu úhlu vůči úhlu referenčnímu. K činnosti algoritmu je třeba tabulka, která v jednom sloupci obsahuje vzdálenosti značek a v druhém sloupci odpovídající číselné vyjádření úhlu ve stupních. Vzdálenosti, které nejsou přímo uvedené v tabulce (většina), je potřeba odvodit z interpolační funkce.

Při výměně kamery je potřeba znovu vytvořit tabulku hodnot za předpokladu, že kamera nebyla zkalibrovaná. Tento způsob detekce úhlu nevyžaduje počáteční kalibraci kamery, co se týče odstranění vad v obraze. Měření úhlu ze vzdálenosti dvou bodů je odolnější vůči deformacím materiálu.

Další možností umístění značek je připevnění jednoho stabilního bodu na nepohyblivou část křídla, druhá značka zůstane na klapce či jiném pohyblivém zařízení. Protože se v tomto případě pohybuje pouze jedna značka, rozdíl ve změně jejich vzdálenosti je největší z uvedených možností.

Ačkoli se nejedná o obecné řešení, které by pokrylo všechny změny úhlu i při posunutí kamery, i metodou měření vzdálenosti 2 bodů lze detekovat stav, kdy se kamera posunula, otočila apod. V případě předchozí kalibrace mírný posun či rotace nevadí, protože se měří vzdálenost v obraze a ne souřadnice. Pokud ale kamera zkalibrovaná nebyla, jejím posunutím se změní stupeň deformace měřeného výřezu a tím se změní i vzdálenosti. Toto nesprávné chování lze detekovat kontrolou, zda se vzdálenosti dostaly mimo požadovaný rozsah. Pokud je vzdálenost pro maximální úhel h a aplikace naměří vzdálenost ještě větší, pak došlo ke změně úhlu pohledu kamery (nebo destruktivní deformaci letounu).

Spolehlivost rozpoznání značek se dá zvýšit použitím několika dvojic. Každá dvojice na základě své tabulky určí úhel. Výsledné 3 úhly jsou aritmeticky zprůměrovány. Tímto se dosáhne nejen větší spolehlivosti, kdy jedna dvojice markerů může být překryta překážkou, ale také přesnosti. Pro n dvojic značek musí existovat n převodních tabulek, protože vzdálenosti jsou vlivem perspektivní projekce u každé dvojice jiné.



Obrázek znázorňuje pozici značek v perspektivní projekci a jejich vzdálenosti

2.7 Interpolace

Při měření úhlu ze vzdálenosti dvou bodů je potřeba interpolovat známé diskrétní hodnoty k tomu, aby bylo možné vyjádřit jakoukoli hodnotu. Interpolační polynomy a interpolační funkce nabízejí možnost proložit diskrétní hodnoty spojitou křivkou.

Věta o existenci a jednoznačnosti interpolačního polynomu:

> Nechť jsou dány body $[x_i, f_i]$, $i \in \mathbb{Z}^+$. Pak existuje právě jeden polynom P_n stupně nanejvýš *n* takový, že $P_n(x_i) = f_i$, $i \in \mathbb{Z}^+$.

2.7.1 Lagrangeův interpolační polynom

Lagrangeův interpolační polynom je numerická metoda pojmenovaná po Josephu Louisi Lagrangeovi. Poprvé byla objevena v roce 1779 Edwardem Waringem. O 4 roky později byla znovu objevena Leonhardem Eulerem.

Interpolační polynom daný body $[x_i, f_i]$, $i \in \mathbb{Z}^+$ sestavíme pomocí polynomu $l_i(x)$ takových, že:

$$l_i(x_j) = \begin{cases} 1 \; pro \; i = j \\ 0 \; pro \; i \neq j \end{cases} \tag{15}$$

$$l_i(x) = \frac{(x - x_0) \cdots (x - x_{i-1})(x - x_{i+1}) \cdots (x - x_n)}{(x_i - x_0) \cdots (x_i - x_{i-1})(x_i - x_{i+1}) \cdots (x_i - x_n)} \tag{16}$$

$$l_0(x)f_0 + l_1(x)f_1 + \cdots + l_n(x)f_n = \sum_{i=0}^{n} l_i(x)f_i \tag{17}$$

Obecnou nevýhodou Lagrangeova polynomu je, že při přidání nebo smazání jednoho jediného prvku se musí celý polynom znovu přepočítat. V problému detekce stavu letounu se počet známých bodů mění pouze při kalibraci. Navíc počet bodů je v rámci jednotek (maximálně desítek), takže i přes kvadratickou časovou složitost algoritmu není zatížení procesoru nevyhovující, pokud jej počítáme pro každý bod zvlášť. Ideální variantou přesto zůstává analyticky předpočítaný polynom. Další nevýhodou je, že v četných případech se interpolační křivka v okolí prvních a posledních bodů značně vzdaluje od analytického řešení. Jistého zpřesnění lze dosáhnout, pokud přidáme další dva body (jeden na začátek, druhý na konec seznamu bodů). Souřadnice těchto bodů můžeme získat lineárně z předchozích 2 bodů. Dva nově přidané body „nasměrují“ křivku polynomu pod očekávaným úhlem na první interpolační bod.

Ukázka algoritmu výpočtu hodnoty bod_y v bodě bod_x v jazyce C++:

```
double interpolaceLagrange(double bod_x, double x[], double y[], int stupen)
{
	double bod_y = 0;
	for (int i = 0; i < stupen; ++i) // pro všechny členy
	{
		double vaha = 1;
		for (int j = 0; j < stupen; ++j)
		{
			if (j != i) // přeskočení i-tého členu
			{
				vaha *= (bod_x - x[j]) / (x[i] - x[j]);
			}
		}

		bod_y += vaha * y[i];
	}
	return bod_y;
}
```

2.7.2 Newtonův interpolační polynom

Tato interpolační technika, kterou objevil Isaac Newton, bývá označována jako metoda poměrných diferencí. Jako výhodnější tvar pro interpolací bodů se jeví následující:

$$P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \\ + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}) \tag{18}$$

Koeficienty a_0 až a_n lze získat dvěma způsoby, z nichž jednodušší a méně pracným je tabulka poměrných diferencí.

Vztah pro poměrnou diferenci prvního řádu:

$$f[x_i, x_{i+1}] = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}, i \in \{0, 1, \dots, n-1\} \tag{19}$$

Poměrné diference vyšších řádů se spočtou pomocí poměrných diferencí řádu nižšího:

$$f[x_i, x_{i+1}, \dots, x_{i+k}] = \frac{f[\,x_{i+1}, x_{i+2}, \dots, x_{i+k}] - f[x_i, x_{i+1}, \dots, x_{i+k-1}]}{x_{i+k} - x_i}, \\ i = 0, 1, \dots, n-k \tag{20}$$

Takto získané poměrné diference odpovídají koeficientům a_1 až a_n v původním polynomu. Je možné ho proto přepsat do následujícího tvaru:

$$P_n(x) = f(x_0) + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \cdots \\ + f[x_0, x_1, x_2, \dots, x_n](x - x_0)(x - x_1)\cdots(x - x_{n-1}) \tag{21}$$

Tento vztah lze upravit postupným vytýkáním podobně jako úprava Hornerovým schématem.

3 Implementace

3.1 Technika

V této kapitole uvádím seznam použitých zařízení a jejich bližší specifikaci.

3.1.1 Použité letadlo

Program byl testován na videu pořízeném z reálného letadla VUT100-131i Cobra s rozpětím křídel 10,2 metrů, délkou 8m. Jedná se o dolnoplošník s vrtulí a jedním motorem v přední části letounu. Bylo vyvíjeno za podpory VUT. Křídlo je celokovové – skládá se z centroplánu a vnějšího křídla. Potah tvoří duralový plech. Klapka je Fowlerova, takže při jejím použití je přistávací rychlost optimální. Protože klapka při vysouvání provádí jak rotační, tak translační pohyb, bylo podle prvního návrhu třeba zjišťovat jak rotační matici, tak translační vektor. Prostor uvnitř křídla je vyplněn vzájemně propojenými integrálními nádržemi o objemu 340 l.

Klapka letounu má 3 stabilní pozice (řízené servomotory)

- 0° – pro let
- 20° ± 1° – pro přistávání
- 40° ± 1° – pro vzlet

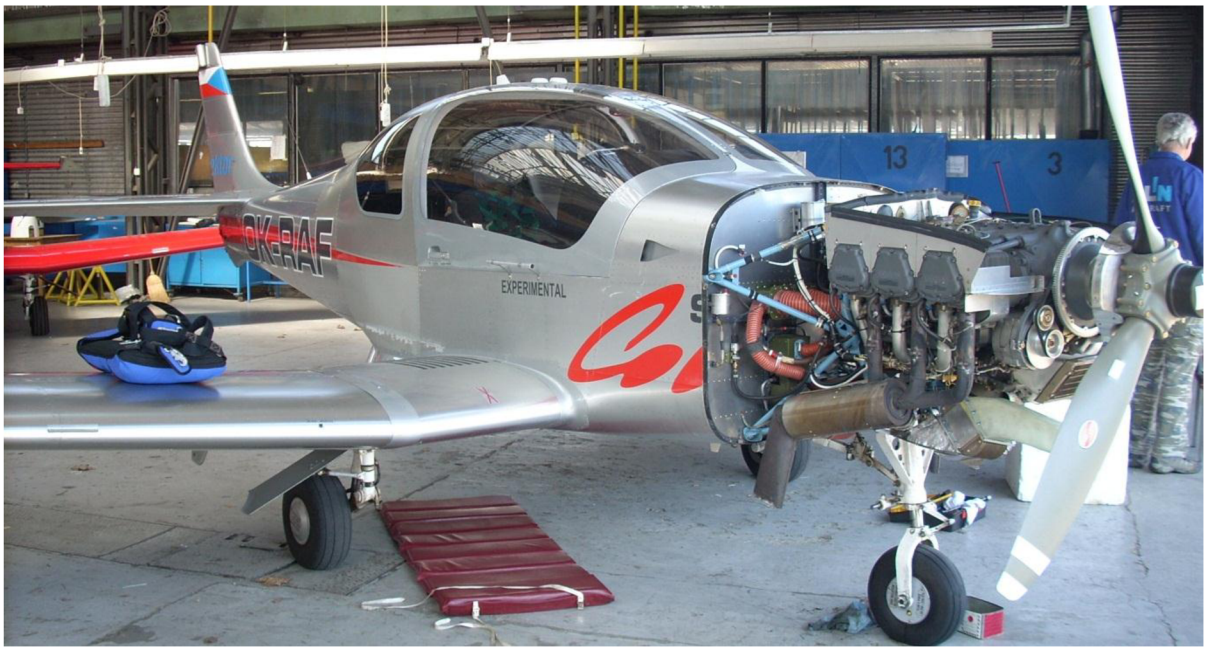
- Rozpětí klapky: 1,835 m
- Plocha klapky: 0,70 m^2

Křidélka mají spojitý prostor možných úhlů a to v intervalu

- β_H: 0 – 25° ± 1° při výkyvu nahoru
- β_D: 0 – 15° ± 1° při výkyvu dolů

- Rozpětí křidélka: 1,735 m
- Plocha křidélka: 0,58 m^2

Celkový rozsah klapky i křidélek je 40° ± 2°, což umožňuje snímat celý rozsah jednou kamerou. Pro snímání vícero veličin (klapka a zároveň křidélko) je potřeba použít větší počet kamer nebo kameru s vysokým úhlem záběru a vysokým rozlišením záznamu.



Obr. 10: VUT100-131i Cobra s odmontovaným krytem motoru při prohlídce

3.1.2 Kamera

Pro záznam videa byla použita klasická SD kamera standardu PAL s progresivním snímkováním a rozlišením 720:576 pixelů. Při snímání videa byl zvolen čas 1/50 s kvůli dostatečné ostrosti. Pro zvýšení přesnosti rozpoznání úhlu byl použit optický zoom. Během natáčení byla kamera pevně připevněna na letadle. Otřesy, které jsou patrné na videu, jsou způsobené pohyby celého letadla a deformací materiálu.

3.2 Výběr softwarových technologií

V zadání bakalářské práce nebyl uveden programovací jazyk ani jiné technologie, kterými má být výsledná aplikace realizována. Před samotným vývojem bylo tedy třeba promyslet, jaké nástroje budou pro danou úlohu nejvhodnější. Protože se jedná o práci s rozpoznáváním obrazu, není třeba psát všechny algoritmy znovu, ale je výhodnější použít už otestované a odladěné algoritmy z volně dostupných knihoven. Na internetu je k dispozici celá řada nástrojů pro detekci hran, pro rozpoznání přímky či úsečky v obraze. Tyto knihovny jsou psány v různých jazycích a je těžké sjednotit jejich kompatibilitu na různých operačních systémech (Linux, Windows). Knihovna OpenCV, která obsahuje velké množství potřebných funkcí (které jsou navíc robustní) a je portována na operační systémy Linux a Windows, byla vybrána pro tento projekt.

OpenCV (Open Source Computer Vision) aktuálně obsahuje více než 500 algoritmů pro real-time zpracování obrazu. Knihovna je šířena pod BSD licencí, která vyžaduje pouze uvedení autora, informaci o licenci a zprávu s upozorněním na zřeknutí se odpovědnosti za dílo. Vývoj projektu počítačového vidění začal v roce 1999, první alfa verze vyšla o rok později. OpenCV existuje v několika variantách programovacích jazyků. Jsou to C, C++, C# a Python. Všechny tyto jazyky jsou široce používané a jejich kompilátory respektive interprety jsou dostupné na obou zvolených platformách. Protože má detektor stavu letounu požadavek na real-timové zpracování, nepřichází v úvahu jazyk C#, kde sice lze psát real-timové aplikace, ale to s ruční zprávou paměti. Při tomhle přístupu se předem generují objekty v paměti, které se ale neuvolňují. Garbage collector tedy

nezatěžuje výkon počítače svou činností. Takový postup je buď paměťově velmi náročný, nebo vyžaduje přesnou zprávu paměti programátorem, kdy nově vznikající objekt využívá už alokované místo v paměti objektu, který už není využíván.

Jako mnohem vhodnější se v začátcích jevil jazyk C. OpenCV je v mutaci tohoto jazyka nejlépe zdokumentovaná. Ostatně firma Intel začala s vývojem knihovny právě v C. V polovině vývoje jsme došli k závěru, že přepsat volání funkcí a metod OpenCV ve verzi C++, by bylo vytvořilo mnohem přehlednější, znovupoužitelnější a kratší kód. Již existující algoritmy byly tedy přepsány do tohoto jazyka. Při dalším vývoji tento krok usnadnil práci, protože některé nové vlastnosti OpenCV 2.0 jsou lépe popsány pro tento jazyk a jejich použití je jednodušší.

Další technologií, kterou bylo potřeba zvážit, byla knihovna pro grafické uživatelské rozhraní (GUI). Mezi uvažované toolkity z důvodu multiplatformnosti patřily wxWidgets, GTK+ a Qt. Kvůli osobním zkušenostem s posledně jmenovaným, byl zvolen jako výchozí implementační nástroj. Qt sice podporuje více programovacích jazyků, ale nejpoužívanějším a nejlépe zdokumentovaným je C++, které se stalo implementačním jazykem celého projektu.

Aplikace Detekce poruchových stavů letounu je tedy postavena na multiplatformních technologiích a je možné je používat na operačních systémech Linux a Windows.

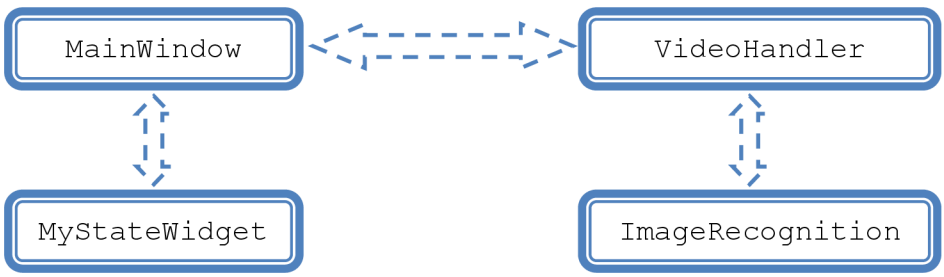
3.3 Objektová modelace

Zvolený programovací jazyk C++ dovoluje vytvořit objektovou stavbu celého projektu. Pro znovupoužitelnost i přehlednost kódu byly odděleny jednotlivé části projektu do objektů, které se starají jen o svou konkrétní funkcionalitu. Výsledný program má následující čtyři části:

- `ImageRecognition` – třída, která zajišťuje samotné rozpoznání v obraze; v projektu má funkci výkonné vrstvy, obsahuje hlavní algoritmy pro detekci chyb letounu
- `VideoHandler` – stará se o přehrávání videa, jeho zastavení, spuštění přehrávání, načasování vykreslení dalšího snímku apod.
- `MainWindow` – presentační vrstva, která vykresluje GUI, ne však video
- `MyStateWidget` – vykresluje video snímky, zachytává události myši nad videem a její chování přeposílá přes presentační vrstvu `MainWindow` `VideoHandleru` a následně `ImageRecognition` vrstvě

V rámci celého projektu je dodržována struktura vícevrstvé architektury. Jestliže tedy presentační vrstva (`MainWindow` či `MyStateWidget`) potřebuje oznámit změnu stavu svého prvku vrstvě výkonné (`ImageRecognition`), pošle ji nejprve vrstvě, která spravuje video (`VideoHandler`) a ta ji teprve přepošle cílové vrstvě (`ImageRecognition`). Toto opatření nabízí jednoduché rozšiřování aplikace v budoucnu.

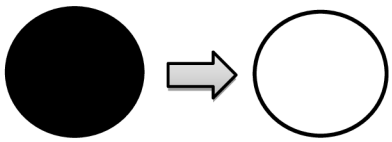
Graf uvádějící popis komunikace mezi jednotlivými objekty:



3.4 Rozpoznání značky

Původní záměr byl vytvořit značku, která by co nejpřesněji určovala střed. Detektor elipsy měl detekovat ohraničení, uvnitř se pak měly detekovat přímky jdoucí do středu. Na jejich průsečíku se nachází střed. Tento tvar byl otestován na letounu, ale výsledek ztroskotal na detekci elipsy. Protože se klapky naklápí v relativně velkém úhlu (zejména u křidélek), značně se změní i velikost vedlejší poloosy elipsy získané projekcí kružnice perspektivní projekcí. Ačkoli tedy aplikace dovoluje nastavit přesnou velikost elipsy, je nutné nastavit interval vzhledem ke všem náklonům klapky či křidélka.

V reálné detekci docházelo k častým problémům s rozpoznáním, kdy detektor elipsy ji detekoval v každé čtvrtině značky. Problém by šel řešit různými filtračními pravidly, ale v případě tohoto projektu, kdy jsou značky snímány z relativně velké dálky, byla zvolena varianta černě vyplněného kruhu, jehož hranová detekce vytvoří potřebný objekt:



Obr. 11: Detektor hran vytvoří z kruhu kružnici

3.5 Detekce úhlu

Nejprve byla implementována detekce úhlu jako získání rotační matice ze změny homografie objektu v perspektivní projekci. Protože klapka křídla není rovina, ale je částečně vypouklá, bylo možno sestavit 3D model středů značek umístěných v prostoru. Jako referenční model byla zvolena skupina pěti markerů, což je dostatečně velký počet pro zjištění změny homografie ve všech směrech a přitom se může stát, že některá ze značek není vidět či není správně detekována, protože minimální počet referenčních bodů v OpenCV algoritmu POSIT je 4. Jak ale ukázala praxe, zjišťování natočení klapky bylo velmi nepřesné. I přes kalibraci kamery docházelo k častým chybám, protože pozice značek je téměř v jedné rovině. Počáteční předpoklad, že mírné zakřivení klapky bude dostatečným, se ukázal jako milný. Pokud je objekt v rovině, jeho homografie odpovídá více rotačním maticím a translačním úhlům. Docházelo tedy k časným výkyvům (např. změnám znaménka) i při nepatrném natočení.

Jako mnohem výhodnější se ukázala metoda pro detekci náklonu objektu ze vzdálenosti dvou bodů. Nejen, že výpočet není tak náročný, ale přesnost výrazně přesahuje metody pro zjišťování rotační matice. Problémem této metody se stalo získání tabulek hodnot, jejichž body se interpoluje polynom, do kterého se dosazují aktuální hodnoty vzdálenosti. Protože jsou klapky naklápěny servomotory, nebylo možné získat přesnou tabulku hodnot. Klapky se sice naklánějí spojitě, ale jejich ovládání dovoluje pouze 3 diskrétní hodnoty (u letounu VUT100-131i Cobra jsou to 0°, 20° a 40°). Z podstaty problému však vyplývá, že jestliže jiné hodnoty ani nastavit nelze, tak je není třeba detekovat. V aplikaci jsem se rozhodl detekovat i změnu pozice pro ucelené letové záznamy. Protože však není k dispozici přesnější tabulka hodnot, úhly mezi jednotlivými stupni dopočítávám lineárně. Pokud je tedy aktuální vzdálenost klapky přesně v průměru vzdáleností dvou sousedních stavů, pak je úhel dopočítán jako průměr úhlů, které těmto stavům odpovídají.

Jiná situace nastává u křidélka, které je ovládáno spojitě. Při letu navíc není využíváno v celém svém rozsahu, ale nejčastěji se vyrovnává horizontální poloha vychýlením křidélka v řádu jednotek stupňů (jiná situace je ovšem v akrobatickém létání). Obecně je tedy třeba nejpřesněji detekovat mírné vychýlení ±5°. Pro přesné nastavení programu je potřeba kalibrační video, kde jsou

známé snímky a aktuální úhel (kamera vidí poměry vzdáleností mezi body a úhel zadá uživatel z předchozího měření např. úhloměrem). Takto získanou tabulku hodnot je potřeba zadávat pouze jednou, program ji pak udržuje v paměti nebo ji zle načíst z textového souboru.

V paměti je tabulka řešena jako vektor asociativních polí. Vektor proto, že pro každou dvojici značek je generována tabulka zvlášť. Asociativní pole je výhodné, pokud se měřené pohyblivé zařízení často naklání na referenční úhel (např. 0°). V tomto případě program pouze zjistí odpovídající úhel na základě indexu, kterým je vzdálenost.

3.6 Popis programu

Protože program zpracovává velké množství dat a nastavení, které by bylo těžké zadávat jako parametry nebo jinou formou z příkazové řádky, bylo vytvořeno grafické uživatelské rozhraní. Základem aplikace je přehrávání videa. K dispozici jsou ovládací prvky typu „spusť“, „pauza“, „stop“… Části aplikace s nastaveními jsou myší uchopitelná a přetáhnutelná, takže si uživatel může plochu přizpůsobit podle svých představ. Pokud zrovna nejsou potřeba, je možné je zavřít. Při opětovné potřebě je lze na uživatelem nastaveném místě zobrazit z menu programu.

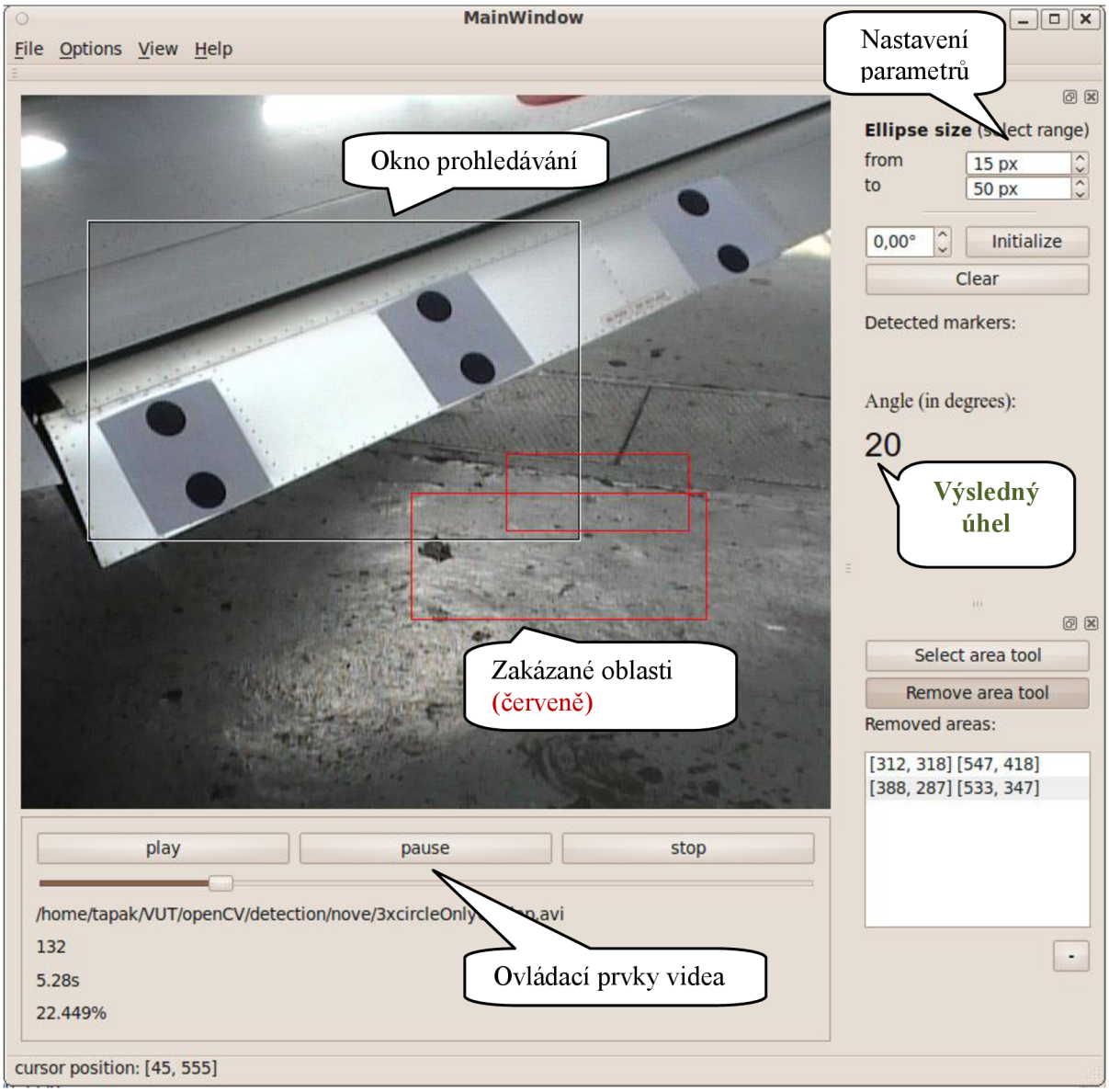
Mezi nejdůležitější části patří nastavení parametrů detekce. Program prohledává pouze ty značky, které splňují požadavky na velikost – od a do dané velikosti (udávaných v obrazových pixelech). Všechny značky, které nesplňují podmínku, jsou z prohledávání vyloučeny. Pokud i tak dochází k rozpoznání nesprávných značek, zle použít mechanismus pro přesný výběr požadované oblasti prohledávání. Při vybrání tlačítka „select area tool“ je možné myší vybírat oblast v obraze. Tím se oblast prohledávání zmenší a zároveň klesne i nárok na výkon počítače. Pokud i v rámci vybrané oblasti dochází k vyhledání nežádoucích značek, je k dispozici nástroj „remove area tool“, který slouží k výběru oblastí, které nemají být prohledány. Oblasti zakázaného prohledávání jsou vizualizovány červeným obdélníkem přímo ve videu, ale jejich souřadnice lze prohlížet v seznamu „removed areas“. Odtud je možné jednotlivé záznamy vymazat pomocí tlačítka „-“. Vyloučeny jsou všechny značky, které mají nenulový průnik s alespoň jednou zakázanou oblastí. Díky těmto nástrojům lze celkem přesně vybrat oblast zájmu. Program disponuje filtračními pravidly pro značky. Protože je kamera statická, daná perspektivní projekce dovoluje vyloučit z prohledávání značky, které mají výšku v určitém poměru k šířce. Za základní poměr byl zvolen následující: výška:šířka » 100:50. Všechny značky, které mají výšku dvojnásobnou a vícenásobnou k šířce, algoritmus vylučuje.

Další možností programu je nastavení tabulky vzdáleností a jim odpovídajícím úhlům. K tomu slouží tlačítko „Initialize“. Po jeho zmáčknutí se do paměti uloží hodnoty vzdáleností právě nalezených značek. Pro přesné nastavení je potřeba kalibrační video, kde se měřený objekt na chvíli zastaví v určitém známém bodě. V tomto místě videa je potřeba přesně vybrat uvažované značky, a protože je známý aktuální úhel natočení z kalibračního videa (např. měřením úhloměrem apod.), je možné údaj zanést do tabulky vzdáleností a úhlů. K zadání aktuálního úhlu slouží editační políčko vedle tlačítka „Initialize“. Údaj je potřeba zadat v úhlových stupních.

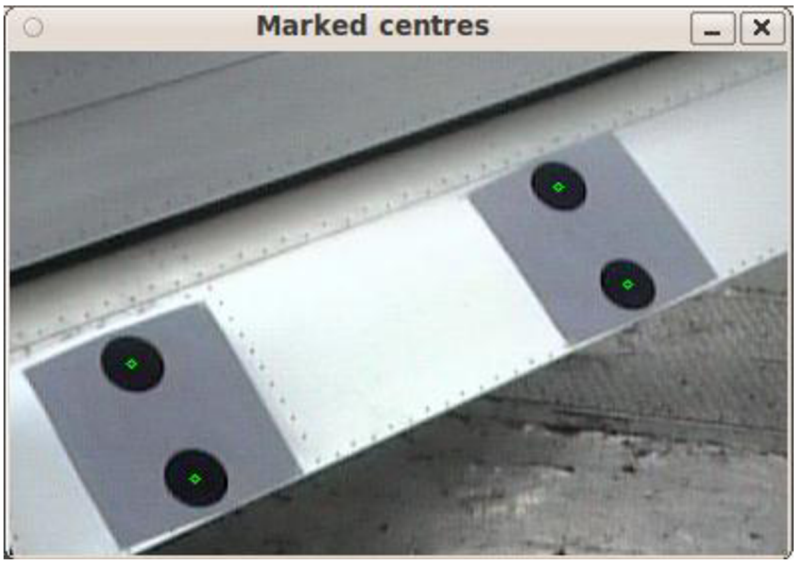
Spočítaný úhel je vizualizován přímo na ploše programu, ale je možné jej ukládat jako proud hodnot do souboru; ten pak následně zpracovávat jako letové hodnoty.

Uložit do souboru lze i tabulku se vzdálenostmi a úhly. Položka je přístupná z menu jako „File – Save lookup table“. Pro opětovné použití lze tabulku z textového souboru načíst rovněž z menu (File – Open lookup table).

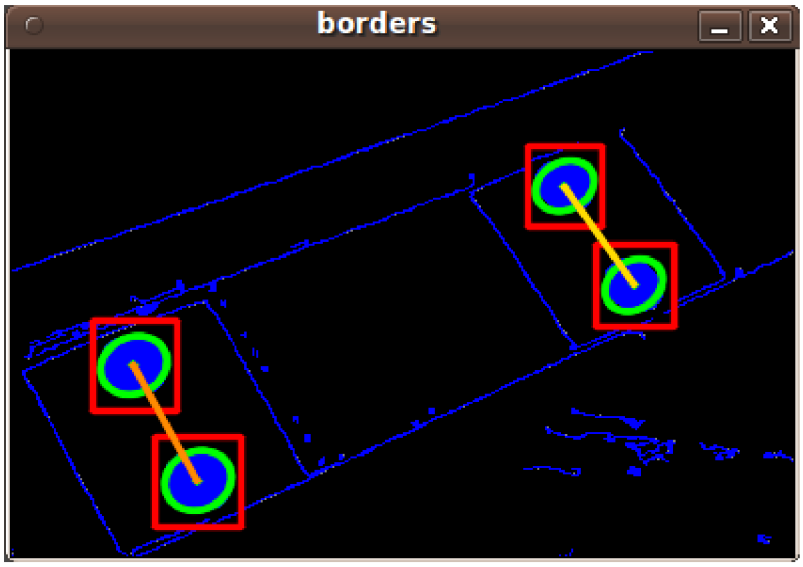
Program lze přepínat buď mezi přehráváním videa z již existujícího souboru, nebo je zpracováváno proudové video přímo z kamery. Aplikace dokáže přehrávat proudové video nebo video uložené v kontejneru *.avi (obě nekomprimované).



Obr. 12: Výsledná podoba aplikace



Obr. 13: Klapka s vyznačenými středy značek



Obr. 14: Pomocné okno s vyznačenými okraji značek a vykreslenou vzdáleností

3.7 Přesnost

Přesnost výsledného úhlu závisí na mnoha faktorech a je těžké je obecně vyjádřit jedním vztahem. Základem je precizní určení středu značky. Pro rozpoznání značky je nutné detekovat hrany – ty jsou klíčovou záležitostí celého projektu. Pokud byl nastaven v kameře dlouhý snímací čas, aplikace se stala nepoužitelnou. Proto je nutné mít dostatečně nízký čas (alespoň 1/50 s), aby každý snímek i při třesoucí se kameře byl ostrý a bylo možné bezpečně detekovat hrany. Jako velmi nevhodné pro tento typ problému jsou zařízení s prokládaným snímkováním. V každém taktu je potřeba získat celý ostrý snímek (progresivní snímkování). Přesnost lze zvýšit optickým zoomem.

Nepodařilo se vypracovat statistickou analýzu získanou ze stovek dat, protože testovací letadlo, na kterém byla práce zkoušena, nemá přesný úhel náklonu ve svých letových datech. Byly použity tedy ručně naměřené hodnoty, které vytvořily samotnou referenční tabulku, takže hodnoty přesně odpovídají.

4 Závěr

Cílem této práce bylo vytvořit aplikaci, která by byla schopná pomocí kamery rozpoznat stav pohyblivých součástek letounu. Požadavek na dobu zpracování byl ten, aby se proud obrazových dat mohl zpracovávat v reálném čase. Účelem aplikace je zvýšit bezpečnost létání přidáním bezpečnostních prvků.

Protože podle zadání nebyly požadavky na systém ani jiné softwarové technologie, byly zvoleny vývojové nástroje C++, Qt toolkit a OpenCV. Tyto technologie zajišťují multiplatformnost a zpracování v reálném čase a přitom umožňují uživatelský komfort typu objektové modelace. Veškerý vývoj byl prováděn pod operačním systémem Linux (Ubuntu). Nebyla použita jediná specifická vlastnost tohoto systému, proto je program obecně přenositelný na jakoukoli platformu podporovanou ze strany Qt toolkitu a OpenCV.

Podařilo se implementovat aplikaci, která podle zadaných parametrů dokáže určit natočení klapky či jiného pohyblivého zařízení. Prvotní myšlenkou bylo získání rotační matice značek z perspektivní projekce a znalosti modelu. Jedná se sice o obecný postup, který potřebuje pouze kalibraci kamery a jeden referenční úhel, ale jako přesnější a rychlejší se ukázala varianta s předem uloženou tabulkou hodnot vzdáleností a jim odpovídajícím úhlům. Tato myšlenka vychází z vlastnosti perspektivní projekce – vzdálenosti v 2D obraze mezi body se při rotaci 3D objektu mění. Varianta určování vzdáleností je odolná vůči deformacím materiálu.

Ideálním výsledkem této práce by byla hardwarová součástka nezávislá na systému a jiných knihovnách. Nabízí se možnost implementace pomocí hradlového pole. Nejsložitější částí v takovém případě se zdá implementace detektoru hran a značek. Vytvoření takové součástky je však nad rámec této práce a je možné ji uvažovat jako další rozšíření.

Literatura

[1] HLAVÁČ V.: *Hledání hran* [online]. Praha: 2010, [cit. 2010-01-02]. Dostupné na URL: <http://cmp.felk.cvut.cz/~hlavac/Public/TeachingLectures/DetekceHran.pdf>

[2] HLAVÁČ V.: *Jasové a geometrické transformace* [online]. Praha: 2010 [cit. 2010-05-10]. Dostupné na URL: <http://cmp.felk.cvut.cz/~hlavac/TeachPresCz/11DigZprObr/18BrightGeomTxCz.pdf>

[3] BRADSKI G., KAEHLER A.: *Learning OpenCV*. O'REILLY, Sebastopol, 2008. ISBN 978-0-596-51613-0

[4] FAJMON, B.; RŮŽIČKOVÁ, I.: *Matematika 3*. Brno : [s.n.], 2005. Aproximace funkcí, s. 255. Dostupné z WWW: <http://www.umat.feec.vutbr.cz/~hlavicka/skripta/matematika3.pdf>.

[5] CHAPRA, S. C., CANALE, R. P.: *Numerical Methods for Engineers: With Software and Programming Applications*. Fourth Edition, McGraw-Hill, New York 2002.

[6] VITÁSEK, E.: *Numerické metody*. Praha, SNTL 1987.

[7] THEODORIDIS, S.; KOUTROUMBAS, K.: *Pattern recognition*. Second edition. USA : Academic press, 2003. 689 s. ISBN 0-12-685875-6.

[8] DEMENTHON, Daniel F.: *Model Based Object Pose in 25 Lines of Code* [online]. Maryland : University of Maryland, [2008]. 30 s. Referát. University of Maryland. Dostupné z WWW: <http://www.cfar.umd.edu/~daniel/daniel_papersfordownload/Pose25Lines.pdf>.

[9] VUT 100 Cobra. In Letecký ústav - projekty. [s.l.]: [s.n.], [2008] [cit. 2010-05-13]. Dostupné z WWW: <http://lu.fme.vutbr.cz/projekty.php?projekt=vut100&full=vv>.

[10] Hledání parametrického popisu objektů pomocí Houghovy transformace. In Houghova transformace. [s.l.] : [s.n.], 2008 [cit. 2010-05-13]. Dostupné z WWW: <http://cmp.felk.cvut.cz/cmp/courses/ZS1/Cviceni/cv5/hough.htm>.

[11] Digimanie [online]. 7.3.2003 [cit. 2010-04-16]. Digitální fotoaparát V: Světlocitlivé snímací prvky. Dostupné z WWW: <http://www.digimanie.cz/art_doc-D80A41BB4E82A7AFC1256CC60031E152.html>.

[12] Digimanie [online]. 7.3.2003 [cit. 2010-04-16]. Fotomobily: snímací čipy CMOS vs. CCD. Dostupné z WWW: <http://www.digimanie.cz/art_doc-67BCCD2DF7A9F53EC125763F0044663D.html>.

Seznam příloh

Příloha 1. CD se zdojovými kódy a záznamem z letadla