ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE, FAKULTA ELEKTROTECHNICKÁ, KATEDRA KYBERNETIKY

Inteligentní robotika

VIRTUÁLNÍ PRACOVIŠTĚ ROBOT KOMÁR 2

TŘETÍ ETAPA

Bc. Martin Makovička (makovma2@fel.cvut.cz)
Bc. Václav Krtička (krticva1@fel.cvut.cz)

11.5. 2011

1. Úvod

Projekt, kterého se tato zpráva týká, se zabývá autonomně řízeným robotem vybaveným USB kamerou, kterou je schopen otáčet v elevaci a azimutu pomocí Pan-Tilt jednotky [1]. Přímo před tímto robotem se nalézá obrazovka, na které se zobrazuje na bílém pozadí pohybující se černý čtverec simulující pohyb letícího komára. Cílem je, aby robot dokázal zmíněný černý čtverec identifikovat a sledovat v záměrném kříži určeném středovou osou kamery.

Ovládání robotu se provádí pomocí počítače, na který se přistupuje přes vzdálenou plochu na operační systém Windows XP. Zde jsou instalovány veškeré potřebné ovladače pro řízení Pan-Tilt jednotky a USB kamery včetně dalšího relevantního softwarového vybavení, jako je především Matlab.

2. Rozbor problému

2.1 Vybavení

Robot Komár je komplexní systém sestávající se z Pan-Tilt jednotky Megarobot a USB 2.0 kamery Chameleon [1].

Pan-Tilt jednotka je určena k otáčení na ní umístěné kamery ve směru azimutu a elevace. Tyto pohyby zajišťují dva servomotory AI-701 od firmy Megarobotics. Pohony jsou řízeny prostřednictvím sériové sběrnice RS-232, která je dále připojena k řídícímu počítači.

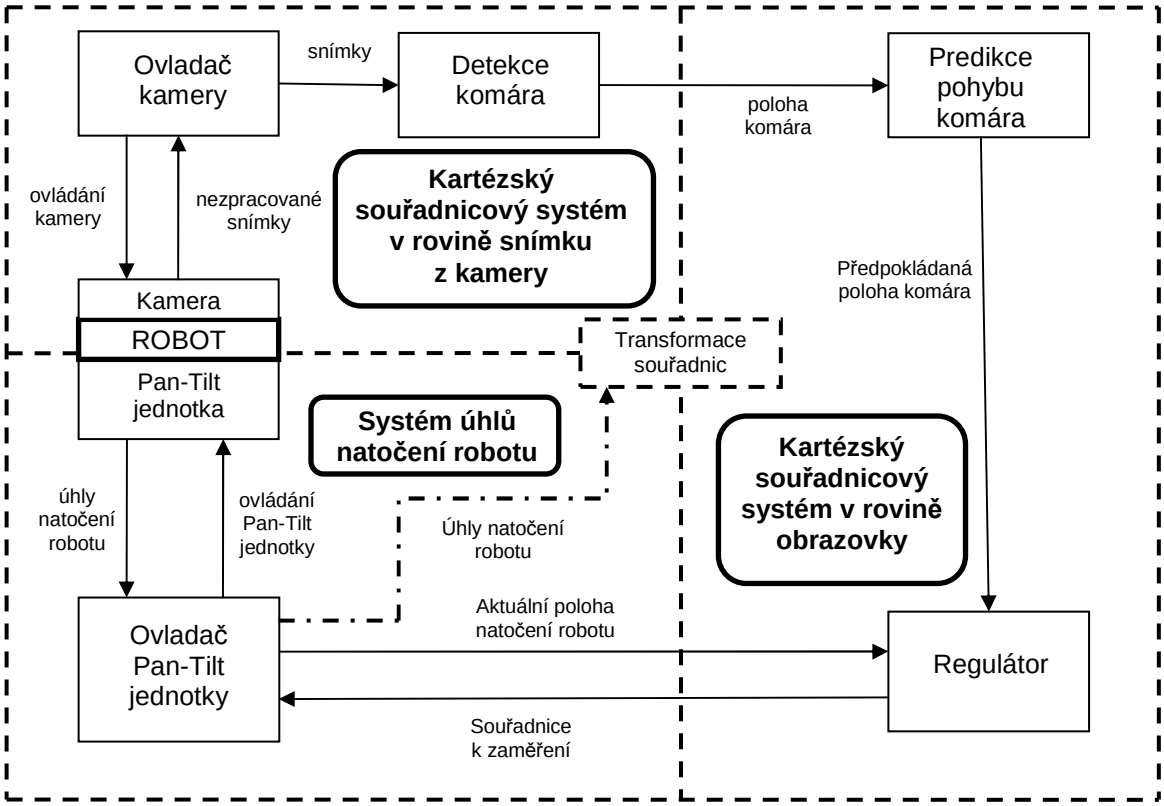
Digitální kamera Chameleon umístěna na výše zmíněné Pan-Tilt jednotce je k řídícímu počítači připojena prostřednictvím rozhraní USB 2.0.

Řídící počítač je přístupný prostřednictvím funkce vzdálená plocha v nainstalovaném operačním systému Windows XP. Softwarovým vybavením jsou mimo jiné Matlab R2010b a ovladače k Pan-Tilt jednotce a digitální kameře.

2.2 Návrh postupu řešení

Řízení Robotu Komár jsme začali programovat pomocí softwarového nástroje Matlab bez použití integrovaného nástroje Simulink. Program je psán metodou objektového

programování. Řízení zaměřování komára bude probíhat pomocí opakující se smyčky, ve které postupně proběhnou všechny relevantní funkce, jejichž provázání je znázorněno na obr. 1.



obr. 1: Schéma návrhu řízení Robotu Komár.

Nejdříve se získá sekvence snímků z kamery prostřednictvím objektu *Ovladač kamery* a s těmito snímky související aktuální natočení kamery získané objektem *Ovladač Pan-Tilt jednotky*. V těchto snímcích se pomocí *Detekce komára* zjistí kartézské souřadnice pozice komára, které se za spolupráce příslušných úhlů natočení kamery převedou *Transformací souřadnic* na kartézské souřadnice pozice komára odpovídající rovině kamerou sledované obrazovky. Z této sekvence souřadnic se pak v *Predikci pohybu komára* predikuje budoucí pozice komára, která je poslána do *Regulátoru*. Zde se za součinnosti známých kartézských souřadnic v rovině obrazovky, na které aktuálně míří kamera, vypočítají cílové kartézské souřadnice záměru kamery v rovině obrazovky. Tyto souřadnice se pak *Transformací souřadnic* převedou na úhly natočení robotu poslané zpět *Ovladači Pan-Tilt jednotky*, která zajistí otáčení kamery směrem na komára. Takto se celý děj nepřetržitě opakuje.

Očekává se, že budou vznikat transportní zpoždění zapříčiněná mimo jiné především zpracováváním obrazu z kamery a převodem souřadnic. Dále pak jistě budou vznikat poziční odchylky mezi snímkem a aktuálním natočením kamery při snímání zmíněného snímku.

3. Řešení problému

Zde je popsáno řešení každé implementované třídy objektů.

3.1 Ovladač Pan-Tilt jednotky

Pro snadnou práci s motory Pan-Tilt jednotky byl pro ně vytvořen ovladač, který pracuje přímo se souřadnicemi v rovině obrazovky a umožňuje jak nastavení polohy kamery tak i vyčtení polohy aktuální.

Ovládání motorů jsme původně řešili způsobem jako v ukázkovém příkladu na webových stránkách [1], ale postupem času se ukázalo, že zadávání a čtení polohy motorů trvá značnou část doby regulační smyčky a bylo potřeba jej upravit. Nejdříve jsme spojili získání aktuální pozice se zadáním požadované pozice, čímž zpoždění pokleslo zhruba na polovinu. Nejvíce času však zpravidla trvalo vykonávání funkce fread, která čte data ze sériové linky. Tato funkce totiž čeká, až bude přijato dostatečné množství dat, a pak teprve data vrací. Abychom tuto prodlevu efektivně využili, použili jsme pro obsluhu sériové linky callback funkce Matlabu, což nám umožní zavolat funkci fread až v okamžiku, kdy je odpověď motoru již uložena v bufferu PC.

Po zavolání funkce pro zadání polohy do motorů je odeslán příkaz pro první motor a hned poté je běh programu uvolněn k dalším činnostem, další obsluha příkazů na sériové lince je provedena v rámci volání callback funkce. První zavolání callback funkce nastane po odpovědi na příkaz prvního motoru, dojde k uložení polohy motoru a je vyslán příkaz pro druhý motor. Odpověď druhého motoru je zachycena stejným způsobem a vyčtené polohy jsou připraveny k vyčtení regulační smyčkou.

3.2 Ovladač kamery

Kamera je ovládána stejně jako Pan-Tilt jednotka obdobným způsobem, jak je znázorněno na příkladu na týchž webových stránkách, viz [1]. Po inicializaci kamery se následně spustí snímání kamerou, kde se do interního bufferu ukládají jednotlivé snímky. Celkem snímáme asi 18 snímků za sekundu dle dokumentace [3]. V bufferu kamery se postupně hromadí kamerou zachycované snímky, které postupně zaplňují jeho paměť. Snímky z kamery získáváme implementovanou funkcí jednou za regulační smyčku. Při použití této funkce dojde k vrácení nejstaršího snímku a zároveň k jeho vymazání z bufferu. Tímto vyvstává riziko, že by při zpožďování v regulační smyčce mohlo dojít k postupnému získávání starších, a tím méně relevantních, snímků, které by nám mohly rozhodit celkové zaměřování. To je řešeno tím, že po každém přečtení snímku je paměť bufferu vymazána. Kdybychom vymazávali paměť bufferu před získáním snímku, abychom si tím zaručili opravdu nejaktuálnější snímek, pak by docházelo k zbytečným zpožděním, než by kamera tento snímek zachytila.

Obraz z kamery je získáván ve formátu RAW, který následně pomocí implementované funkce „debayerizujeme“ [1] na matici RGB. Tuto pak dále převedeme na šedotónový obraz, který tak sníží náročnost identifikace komára ve snímku.

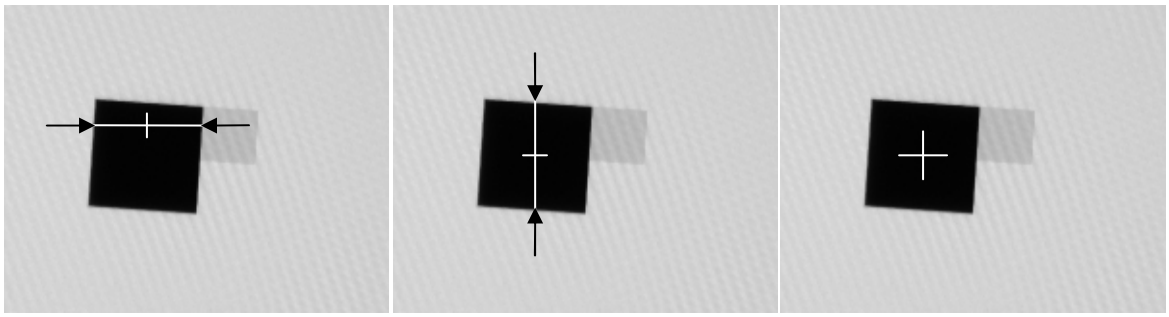
3.3 Detekce komára

Tento objekt je zařazen hned za *Ovladač kamery*, ve kterém je v příchozích snímcích detekován komár. Výstupem je pak příznak existence komára a jeho kartézské souřadnice v rovině daného snímku. Pokud komár není v daném snímku nalezen, pak příznak existence komára je negativní. To znamená, že kamera není v době snímání natočena tak, aby komár byl v jejím zorném úhlu, nebo je zachycen na kraji snímku.

Uvažujme empiricky zjištěné konstanty, jejichž využití bude uvedeno v následně popsaném postupu detekce komára. Mějme hodnoty týkající se jasů pixelů ve snímku jako horní jasová hranice $B_{lim} = 180$ a jasová hranice tmavosti komára $B_{gnat} = 70$. Dále pak hodnoty týkající se velikosti komára jako minimální velikost komára $G_{smin} = 3$, maximální velikost komára $G_{smax} = 80$ a tolerance velikosti komára $G_{stol} = 10$.

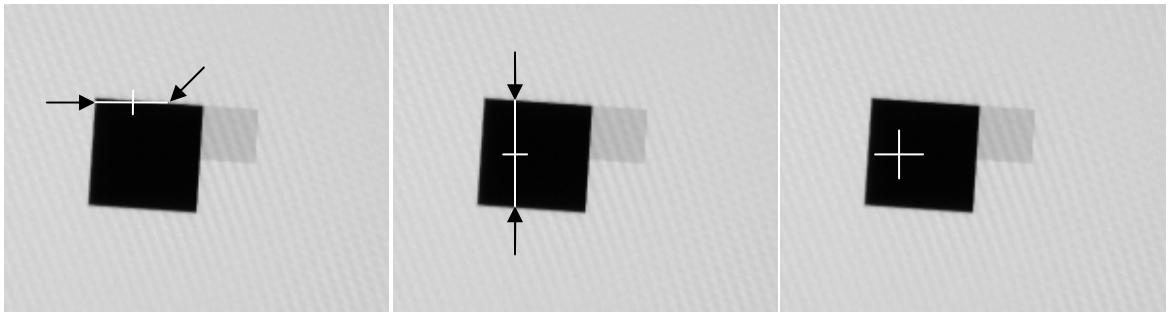
Detekce pracuje v několika fázích. První fází je nalezení nejtmavšího pixelu ve snímku z kamery. Tato fáze může v mnoha případech zkrátit čas hledání komára především, když snímek pochází z takového natočení kamery, že snímá pouze aktivní plochu obrazovky. Pokud je jas nalezeného nejtmavšího pixelu větší než B_{gnat}, pak to znamená, že se komár ve snímku nenalézá. Pokud je tomu však naopak, pokračuje se druhou fází.

V druhé fázi se určí obdélníkové hranice kolem tohoto pixelu dané součtem G_{smax} a G_{stol} na každou ze čtyř stran od tohoto pixelu. Pokud vypočtená hranice sahá za okraj snímku, pak se logicky samotný okraj snímku stává hranicí. V tomto ohraničeném obdélníku se vyhledává vždy po několika vynechaných řádcích (my jsme určili 4) specifický počet pixelů vedle sebe v řádku, které mají úroveň jasu pod B_{lim}. Tento počet se musí vejít do tolerance danou G_{smin} a G_{smax}. Tato sekvence tmavších pixelů musí být z obou stran ohraničena minimálně jedním pixelem světlejším, překračujícím B_{lim}. Jakmile je takováto sekvence v daných tolerancích nalezena, ověřuje se, zda se jedná o komára stejnou metodou ve sloupci, který protíná střed zmíněné řádkové sekvence. Středy těchto sekvencí pak značí střed nalezeného komára, viz obr. 2.



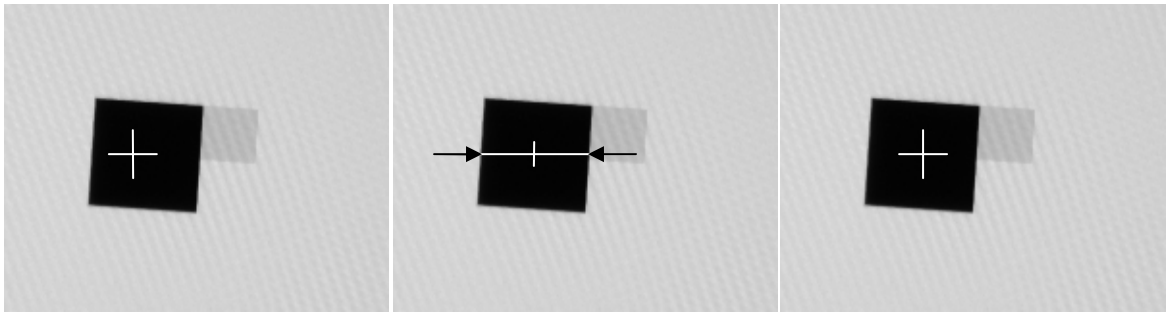
obr. 2: Demonstrace postupu detekce komára a nalezení jeho středu.

Bylo zjištěno, že nepřesnosti zachycování snímků s komárem, kdy komár je zachycen mírně šikmo, může střed komára být špatně identifikován mimo skutečný střed komára v horizontálním směru. Tato chyba je znázorněna na obr. 3.



obr. 3: Demonstrace chybné detekce středu komára v horizontálním směru.

Tento nedostatek jsme napravili opětovným zjištěním sekvence tmavších pixelů tentokrát v řádku odpovídajícímu středu předtím zjištěné svislé sekvence, viz obr. 4.



obr. 4: Demonstrace „dodetekování“ středu komára z chybné detekce, viz obr. 3.

Ve třetí fázi je pak ještě ověřen jas nalezeného středu představujícího detekovaného komára pod B_{gnat}. Toto ověření bylo nutné zavést jako kontrolu, že se jedná skutečně o komára. Vyplývá to z faktu, že pokud není v laboratoři dostatečná tma, může být falešně detekován komár kdekoliv mimo snímanou obrazovku, kde se simulovaný komár zobrazuje. Sekvence tmavších pixelů se totiž může pohybovat náhodně i těsně pod B_{lim}. Tento nedostatek lze ještě dodatečně odstranit omezením prohledávané plochy ve snímku hranicemi v rámci snímané obrazovky s komárem a to na základě aktuálního natočení kamery souvisejícím s daným snímkem. To jsme však z časových důvodů nerealizovali.

3.3 Transformace souřadnic

Abychom byly schopni předpovídat pohyb komára, potřebovali jsme jeho polohu určovat mimo souřadný systém obrazu kamery, aby poloha komára nebyla závislá na pohybu kamery. K určování polohy komára jsme použili souřadného systému určeném rovinou obrazovky s třetí osou kolmou k obrazovce. Počátek souřadnic jsme umístili do středu kamery v okamžiku, kdy míří na střed obrazovky.

Pro určování pozice komára jsme se rozhodli využít toho, že se komár pohybuje pouze v jedné rovině (po obrazovce) a jeho polohu tedy určujeme pomocí dvou souřadnic této roviny. Dle zveřejněných parametrů Pan-Tilt jednotky jsme pomocí Robotic toolboxu v Matlabu sestavili její model, viz obr. 5.

Typ	α	A	φ	D
R	$\frac{\pi}{2}$	0	0	$ElHeight$
R	0	$ElDist$	$\frac{\pi}{2}$	$AzElDist$
R	$\frac{\pi}{2}$	0	0	0
R	$\frac{\pi}{2}$	0	0	0
P	0	0	π	0

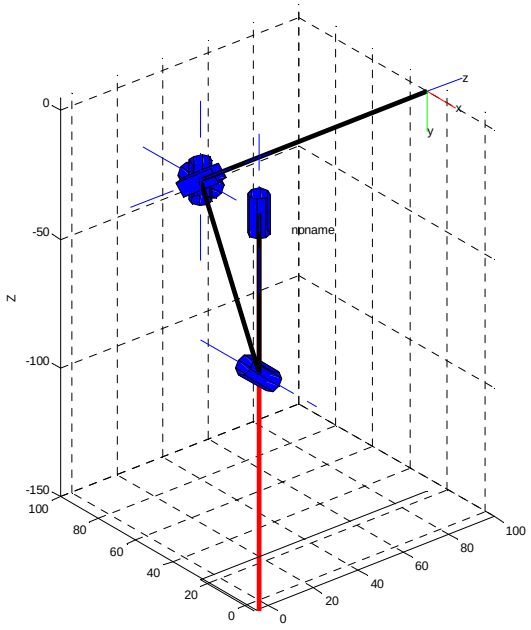
obr. 5: Popis celku dle D-H notace
$ElHeight$ – Vertikální vzdálenost elevační osy od počátku globálního souřadného systému
$Eldist$ – vertikální vzdálenost mezi elevační osou a osou kamery
$AzEldist$ – boční vzdálenost mezi azimutální osou a osou kamery

Parametry modelu jsme určili z mechanických rozměrů kamerového systému, které jsme obdržely spolu se zadáním úlohy. Parametry byly určeny následovně:

ElHeight = -62

Eldist = 62
AzEldist = 31

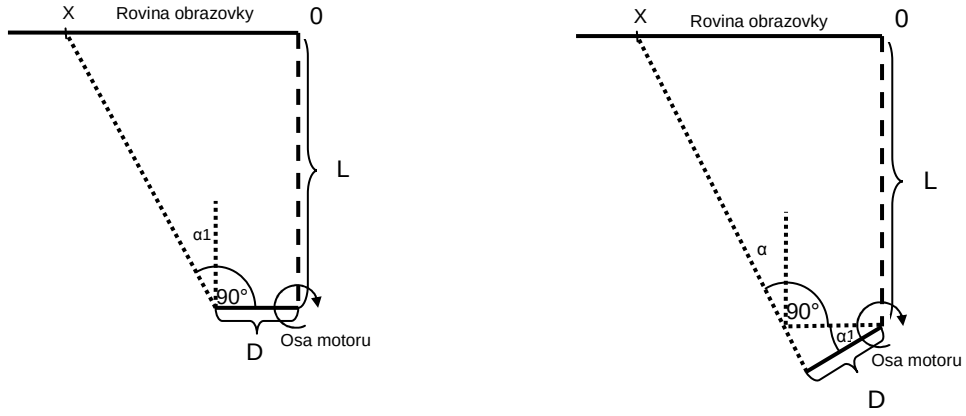
Pro převod ze souřadnic obrazu kamery na souřadnice obrazovky jsme přidali k existujícímu modelu kamery další dva k sobě kolmé rotační klouby, které se nacházejí v optickém ohnisku kamery. Tyto dva klouby simulují odchylky v obraze kamery od jeho středu. Konstanty realizující převod mezi odchylkou od středu obrazu kamery a úhlem virtuálního kloubu jsme určily experimentálně zaměřováním pevného bodu v různých místech obrazovky následovně αx = -4,5E-4 rad/pixel, αy = -4,4E-4 rad/pixel.



obr. 6: Vizualizace použitého modelu kamery v robotic toolboxu.

Výpočty transformací jsme zpočátku prováděli opět pomocí Robotic toolboxu, jakožto nejsnadnějším řešením. Při zkoušení na komárovi jsme však zjistili, že řešení inverzní kinematické úlohy (převod z kartézských souřadnic na úhly motorů) trvá až několik sekund, což je na tuto úlohu příliš pomalé a byli jsme tedy nuceni navrhnout rychlejší výpočet.

Pro výpočet jsme přemístili osu otáčení motoru na konec ramena (D), které je kolmé k ose otáčení, viz obr. 7. S tímto zjednodušením jsme spočítali odhad úhlu α_1 a pro zvýšení přesnosti výpočet zopakovali s ramenem pootočeným o odhadnutý úhel. Při porovnání výsledků této metody s výsledky Robotic toolboxu jsme s překvapením zjistili, že rozdíl hodnot se v oblasti obrazovky i ve vzdálenosti 2 metrů od ní pohybuje v řádech setin procent, zatímco čas potřebný k výpočtu je u nové metody přibližně tisíckrát menší. Tento výpočet jsme použili pro oba motory.



obr. 7: Aproximace inverzní transformace. Vlevo krok 1, vpravo krok 2.

3.4 Predikce pohybu komára

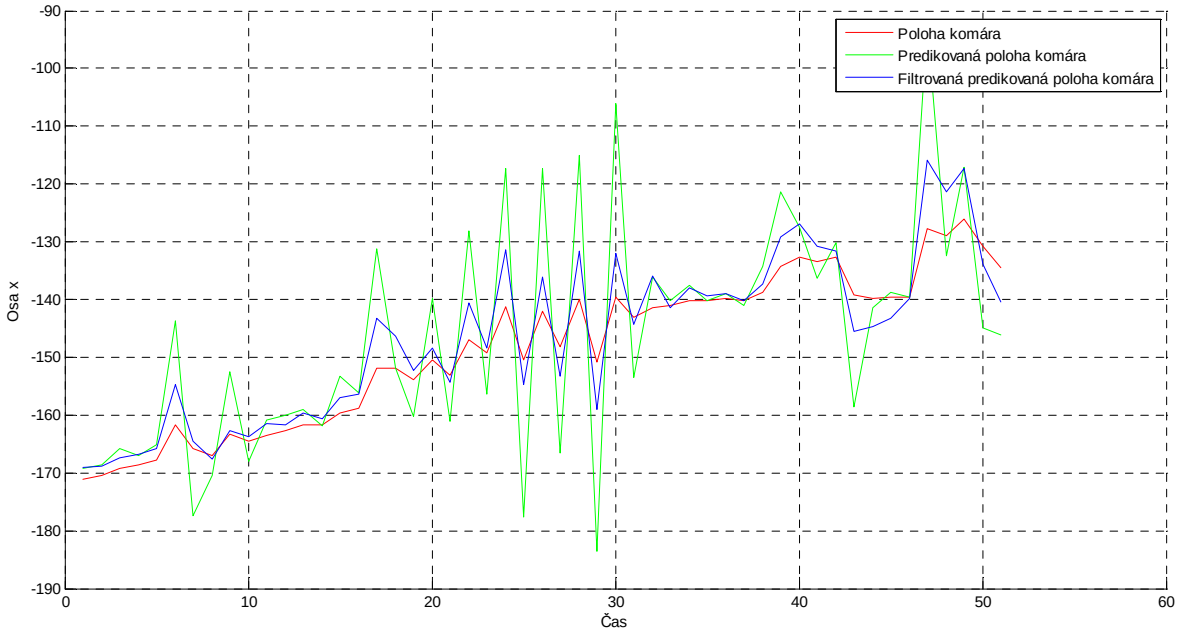
Tento blok slouží k určení budoucí polohy komára ze znalosti jeho předešlých pozic.

K odhadu trajektorie komára jsme původně chtěli průběžně počítat rychlost a zrychlení komára a z něj predikovat jeho polohu. Použití zrychlení se však postupem času ukázalo jako nevhodné, neboť tato složka příliš zesilovala nepřesnosti detekované polohy komára.

Nepřesné zjišťování polohy komára způsobené především nemožností synchronizace zjištění polohy motorů a získání snímku z kamery způsobuje deformaci trajektorie komára, což predikci značně znepřesnilo obr.8. Proto jsme se po několika pokusech rozhodli na výstup predikce přidat FIR filtr, který by měl průměrovat predikovanou polohu a zmírnit tak kmity. Filtr jsme se snažily navrhnout tak, aby měl co nejmenší zpoždění a co dostatečně tlumil výchylky. Ve finální verzi jsme použily filtr třetího řádu s koeficienty [0,5 0,31 0,19]. Zpoždění způsobené filtrací se snažíme alespoň částečně kompenzovat prodloužením výhledu predikce na 1,5 násobek periody regulační smyčky.

Zkoušeli jsme experimentovat i s jiným umístěním filtru (například filtrovat přímo detekovanou polohu komára), ale umístění za predikci se ukázalo jako nejvhodnější.

Predikce pohybu je omezeně schopna odhadovat pohyb komára, i když komár není delší dobu rozpoznán v obraze kamery, což je výhoda v případě přehlédnutí komára, nebo jeho ztracení vlivem krátkého výkyvu regulace. Nicméně jsme dobu predikce pohybu bez detekovaného komára omezili, neboť je pravděpodobné, že komár brzo po ztracení změní směr a predikce by nebyla schopna komára správně trasovat. Původně jsme maximální dobu predikce nastavili na 1s, vzhledem k občasnému zpomalení běhu programu jsme však tuto hodnotu museli zvětšit na přibližně 7s.



obr. 8: Chování detekce komára a predikce při hladké trajektorii komára

3.5 Regulátor

Pro regulaci jsme využili diskrétní verzi PIDf regulátoru, pro nějž jsme použili již hotovou implementaci pro Matlab , kterou jsme vzhledem k občasnému zpomalení běhu programu upravili na proměnnou periodu vzorkování.

Jako referenční (požadovanou) hodnotu regulátoru předkládáme předpokládanou polohu komára v době dalšího měření vstupních dat, tuto polohu porovnáváme s aktuálními souřadnicemi, na které míří kamera.

Při ladění regulátoru jsme se jej snažily nastavit tak, aby spíše kmital, čímž jsme se snažily zajistit rychlejší reakci na změnu směru komára a také zvýšit pravděpodobnost přímého zásahu komára. Nakonec jsme použily následující parametry PIDf regulátoru:

Osa x: P = 0,78; I = 6; D = 0; N = 30

Osa y: P = 0,82; I = 2; D = 0; N = 30

3.5 Další vlastnosti

Pro zlepšení chování zaměřování komára v některých specifických případech jsme regulaci doplnili o několik nestandardních funkcí.

Vzhledem k potížím se sledováním komára při rychlé změně směru pohybu komára u okraje obrazovky jsme se rozhodli omezit výstup predikce pouze na oblast obrazovky a zároveň omezit maximální akční zásah na ±150mm. Po této úpravě se chování regulace při popsané situace zlepšila, další zlepšení nastalo po úpravě algoritmu pro generování trajektorie komára na testovací sestavě.

Další problematickou situací byla ztráta a opětovné objevení komára. Predikce polohy komára je schopna po omezenou dobu předpovídat polohu komára bez aktualizace jeho známé polohy, ale je nastavena tak, že po uplynutí této doby vrací nulovou polohu a kamera by se tedy měla natočit na střed obrazovky, kde je největší pravděpodobnost znovunalezení komára. Problémem však je, že po ztrátě komára se kamera snaží mířit mimo obrazovku a v regulátoru se nekontrolovatelně integruje akční zásah a po změně požadované pozice trvá dlouho, než se neintegrovaná hodnota vrátí do normálu. Proto vždy po ztrátě komára resetujeme PID regulátor, který na změnu polohy zareaguje okamžitě.

Po znovuobjevení komára detekuje predikce pohybu velkou změnu polohy z poslední známé polohy komára před ztrátou kontaktu na nově zjištěnou polohu, která je zpravidla na kraje obrazovky daleko od původní polohy. Predikce tedy zjistí prudkou změnu polohy, kterou ještě zvětší a doba zaměření komára se v lepším případě mnohonásobně zvětší. Abychom tomuto jevu zabránily, uměle zadáme polohu znovuobjeveného komára několikrát za sebou v krátkých časových intervalech do predikce pohybu a do FIR filtru.

4. Závěr

Námi navržený regulátor je schopen celkem spolehlivě udržovat komára v zorném úhlu kamery, problémy nám však dělá přesné zaměření komára, protože pro regulaci potřebujeme co nejrychleji měnit polohy motorů, což ovšem způsobuje zvýšení nepřesností při určení polohy komára v prostoru. Problémem při regulaci také jsou mechanické vlastnosti Pan-Tilt, kde kamera s objektivem pravděpodobně tvoří značnou část hmotnosti zařízení a při prudkých pohybech dochází k výkyvům způsobeným pružností Pan-Tilt jednotky. Další překážkou je velký krok servomotorů, kde jsme schopni zaměřovat pouze s přesností 8mm v rovině obrazovky, a také nepřesná regulace motorů s častým offsetem.

Vzhledem k problémům s přesností určení polohy komára jsme přesvědčeni, že pokud bychom nepoužívaly predikci pohybu, byly bychom schopni dosáhnout lepších regulačních výsledků. Odstranění predikce však v tomto případě nepřipadalo v úvahu, neboť jejím odstraněním bychom odstranili velkou část naší práce a predikce byla zároveň přímo požadována v zadání úlohy.

Oproti problémům s určováním přesné polohy komára náš systém vyniká relativně nízkou periodou regulační smyčky a citlivou detekcí, která je schopna detekovat mnohem menšího komára než detekce ve vyhodnocovacím skriptu. V odevzdávané verzi však citlivost není nastavena na maximu, neboť je zbytečné abychom detekovali menšího komára než je nutné. Snížením citlivosti detekce jsme zároveň snížili riziko detekce falešného komára například mimo obrazovku. Perioda regulační smyčky se většinou pohybuje kolem 70ms, čehož jsme dosáhli především optimalizací práce se sériovou linkou motorů a transformací souřadnic.

Pokud bychom byli schopni určit vzdálenost komára od kamery, šlo by naší regulaci po minimálních úpravách používat v celém okolním prostoru omezeném pouze rozsahem otáčení motorů a přímé viditelnosti. Dále je možné náš systém doplnit o globální senzor pro detekci komára ve větší oblasti, který by mohl kameru navigovat přímo na komára a ta by tak nemusela čekat, až se komár objeví v jejím obrazu.

Pro zlepšení kvality regulace se současným systémem bychom navrhovali vedle kamery umístit laser, který by byl schopen měnit směr zaměření v rozsahu alespoň poloviny plochy obrazu kamery. Díky této úpravě bychom mohli hýbat s kamerou pouze pro hrubé zaměření, čímž by se zmírnil problém s přesným zjištěním polohy kamery, a jemné zaměřování by bylo prováděno nasměrováním laseru, které by mohlo být rychlejší.

Reference

[1] SMUTNÝ, V. *Virtuální pracoviště Robot Komár* [online]. [cit. 2011-05-11], ⟨http://cw.felk.cvut.cz/doku.php/courses/a3m33iro/komar⟩.

[2] *PSD regulátor* [online]. [cit. 2011-05-11], ⟨http://dce.felk.cvut.cz/sari/download/sri/cv11_psd_regulator.pdf⟩.

[3] *Chameleon (USB digital video camera)* [online]. [cit. 2011-05-11], ⟨http://www.ptgrey.com/products/chameleon/Chameleon_datasheet.pdf⟩.