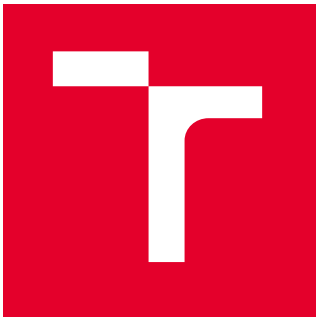


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

METODY STABILIZACE POLOHY DRONU POMOCÍ OBRAZOVÝCH DAT

DRONE POSITION STABILIZATION METHODS USING IMAGE DATA

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — **Ondřej Koukal**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Janoušek**

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Ondřej Koukal ***ID:*** 230100

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Metody stabilizace polohy dronu pomocí obrazových dat

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s problematikou držení přesné polohy bezpilotních letadel typu kvadrokoptéra v 3D prostoru bez signálu GNSS a prostudujte dostupné metody zpracování obrazových dat z kamery.
2. Navrhněte vhodné algoritmy zpracování obrazu, které umožní stabilizaci pozice dronu bez zásahu uživatele na základě detekce okolního prostředí. Zaměřte se především na nízkou výpočetní náročnost pro jednodeskový počítač.
3. Vyberte vhodný kamerový senzor a jednodeskový počítač pro zpracování obrazu přímo na dronu.
4. Implementujte navržené algoritmy pro stabilizaci polohy dronu s řídící jednotkou Pixhawk, popište použité knihovny a jejich funkci.
5. Ověřte funkčnost vytvořeného řešení provedením sady testovacích letů v různých typech reálného prostředí. Vyhodnoťte vlastnosti a spolehlivost experimentů.

DOPORUČENÁ LITERATURA:

Aasish, C. & Ranjitha, E. & Ridhwan, U. & Raj, S. & Jemi, L.. (2015). Navigation of UAV without GPS. Proceedings of 2015 International Conference on Robotics, Automation, Control and Embedded Systems, RACE 2015. 10.1109/RACE.2015.7097260.

Wang, Junjue & Feng, Ziqiang & Chen, Zhuo & George, Shilpa & Bala, Mihir & Pillai, Padmanabhan & Yang, Shao-Wen & Satyanarayanan, Mahadev. (2019). Edge-Based Live Video Analytics for Drones. IEEE Internet Computing. 23. 27-34. 10.1109/MIC.2019.2909713.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 22.5.2023

Vedoucí práce: Ing. Jiří Janoušek

doc. Ing. Václav Jirsík, CSc.
předseda rady studijního programu

ABSTRAKT

Tato bakalářská práce se věnuje návrhu systému pro stabilizaci bezpilotního letadla. Účelem tohoto systému je na základě obrazových dat ze dvou kamer v reálném čase určovat přesnou polohu bezpilotního letadla v prostoru. V teoretické části jsou popsané metody zpracování obrazu a algoritmus pro určení polohy bezpilotního letadla. V praktické části je popsána realizace a testování systému.

KLÍČOVÁ SLOVA

UAV, dron, strojové vidění, detekce příznaků, OpenCV

ABSTRACT

This bachelor's thesis focuses on the design of an unmanned aircraft stabilization system. The purpose of this system is to determine the exact position of the unmanned aircraft in space based on image data from two cameras in real time. The theoretical part describes the image processing methods and the algorithm for determining the position of the unmanned aircraft. The practical part describes the implementation and testing of the system.

KEYWORDS

UAV, drone, computer vision, feature detection, OpenCV

KOUKAL, Ondřej. *Metody stabilizace polohy dronu pomocí obrazových dat* . Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2023, 51 s. Bakalářská práce. Vedoucí práce: Ing. Jiří Janoušek

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Ondřej Koukal
VUT ID autora:	230100
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Metody stabilizace polohy dronu pomocí obrazových dat

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Jiřímu Janouškovi za odborné vedení, konzultace, trpělivost, podnětné návrhy k práci a poskytnutí prostředku nutných k implementaci a testování práce.

Obsah

Seznam obrázků

Úvod

Převážná většina bezpilotních letadel využívá pro určování vlastní polohy přijímač GNSS. Ten však nelze využít například v uzavřených prostorech a je velmi citlivý na rušení signálu.

Cílem této práce je navrhnout systém, který je schopný určit polohu bezpilotního letadla na základě obrazových dat ze dvou kamer. Je pro to využitý přístup založený na sledování referenčních bodů v prostoru. Tento systém by měl být schopný nahradit přijímač GNSS bez nutnosti zásahu do firmwaru bezpilotního letadla.

V práci je popsané základní rozdělení metod pro určování polohy z optických dat, teoretická část práce je zaměřená na algoritmy využitelné pro zvolenou metodu určování polohy.

Dále je v této práci popsaný software a hardware využitý pro testování a určený pro implementaci systému. V posledních dvou kapitolách je popsaná implementace systému a výsledky jeho testování.

1 Metody určování polohy z optických dat

1.1 SLAM

SLAM (z angl. Simultaneous localization and mapping) pokročilý přístup k řešení problému lokalizace dronu za pomocí obrazových dat. Cílem tohoto přístupu je v reálném čase z naměřených dat aktualizovat 3D mapu okolí a následně určit polohu dronu v rámci této mapy. Tuto mapu lze dále využít například pro předcházení kolizím a autonomní navigaci. Tyto metody dosahují velmi dobrých výsledků zejména při pohybu ve dříve zmapované oblasti. Nevýhodou tohoto přístupu je velmi vysoká výpočetní náročnost.[1]

1.2 Systémy založené na předem známém okolí dronu

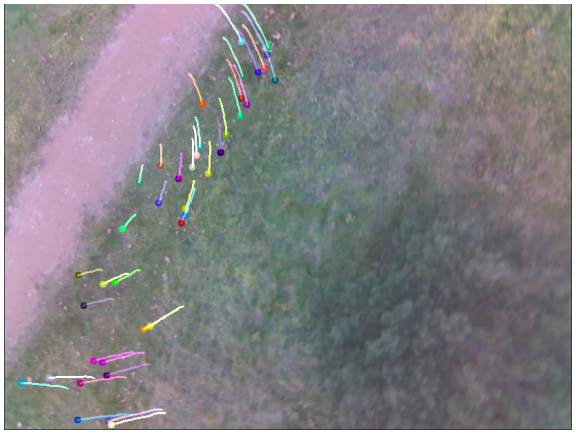
Pro využití v kontrolovaném prostředí lze pro určení polohy dronu využít známé informace o prostředí. V praxi to typicky znamená využití dobře detekovatelných značek na předem určených místech. Polohu dronu lze poté pomocí těchto značek relativně jednoduše triangulovat. Tyto systémy jsou velmi spolehlivé, nelze je však využít v předem nezmapovaném prostředí.[2]

1.3 Vizuální odometrie

Dalším přístupem pro určování polohy dronu je tzv. optická odometrie. Tento přístup spočívá v odhadu pohybu dronu na základě informací z po sobě jdoucích snímků. Oproti metodám SLAM je obvykle méně výpočetně náročná, a může tedy dosáhnout dobrých výsledků i na méně výkonném hardwaru.[1]

1.3.1 Přímá vizuální odometrie

Přímá vizuální odometrie vychází ze zdánlivého pohybu bodů v obrazu mezi snímky. Z tohoto zdánlivého pohybu bodů lze pak odhadnout rychlost a směr pohybu dronu. Poloha dronu pak může být určena jako suma pohybů dronu mezi jednotlivými snímky.[2]



Obr. 1.1: Zdánlivá trajektorie vybraných bodů určená metodou Lucas-Kanade.

1.3.2 Vizuální odometrie založená na příznacích

Tato metoda je založená na detekci výrazných, dobře sledovatelných bodů v obrazových datech. Poloha dronu je pak určována na základě odhadu relativní polohy vzhledem k těmto bodům. [2]

Obr. 1.2: Příznaky detekované metodou ORB.

2 Metody získávání obrazových dat

2.1 Mono kamera

Nejvíce dostupným zdrojem optických dat je kamera s jedním snímačem. Z obrazových dat z jedné kamery je velmi obtížné určit vzdálenost snímaných bodů od optického ohniska kamery, proto je vhodné ji použít v kombinaci s jiným senzorem.[2]

2.2 Stereo kamera

Obrazová data ze dvou různých snímačů umožňují relativně jednoduché určení polohy snímaného bodu. Nevýhodou oproti mono kameře je dvojnásobný datový tok, hmotnost a spotřeba energie. Konstrukční řešení systémů se stereokamerou je také kvůli nárokům na minimální vzájemný pohyb snímačů a velkou vzdálenost mezi snímači relativně komplikované.[2]

2.3 Senzory vzdálenosti

Do této kategorie spadá například lidar, radar, laserové scannery. Senzory tohoto typu s dostatečným rozlišením a rozsahem měřitelných vzdáleností pro samostatné použití jsou extréme nákladné, levnější varianty jsou ale vhodné pro doplnění jiných systémů.[2]

3 Návrh systému

Pro určování polohy využívám systém založený na bodových příznacích. Tento přístup spočívá v hledání výrazných bodů v obrazových datech. Ke každému výraznému bodu je přiřazena jeho poloha, při nalezení stejných výrazných bodů ve snímcích, které po sobě následují, pak lze určit informace o pohybu dronu. Tento přístup umožňuje za cenu vyšší výpočetní náročnosti vyšší přesnost a spolehlivost než přímá vizuální odometrie.

Pro přesné určení polohy výrazného bodu (referenčního bodu) z obrazu je nutné jej nasnímat minimálně ze dvou různých bodů. Toho je možné docílit i mono kamerou, tento přístup je však relativně komplikovaný. Jako zdroj obrazových dat jsem proto zvolil stereo kameru. Snížení výpočetní náročnosti a nepřesnosti odhadu polohy způsobené přítomností druhého snímače umožňuje využití výrazně méně výkonného počítače. Snížení ceny, hmotnosti a spotřeby energie počítače více než vyvažuje přidanou cenu, hmotnost a spotřebu energie stereo kamery oproti mono kameře.

Rozhodl jsem se systém navrhnout tak, aby nebyl vyžadován zásah do firmwaru řídicí jednotky. Úprava softwaru řídicí jednotky by rozšířila možnosti zpracování polohy, ale značně by omezila praktickou využitelnost systému. Dále jsem se rozhodl systém navrhnout pouze pro určovaní umístění dronu v prostoru. Zvolená metoda sledování bodových příznaků výpočet informací o náklonu dronu sice umožňuje, nicméně tyto informace je obvykle schopná poskytnout s výrazně větší přesností řídicí jednotka na základě dat z akcelerometru a gyroskopu.

4 Předzpracování obrazových dat

4.1 Změna měřítka

Pro účely této práce je změna měřítka užitečná primárně pro snížení objemu zpracovaných dat a urychlení výpočtů. Dále jsou popsané dvě nejméně výpočetně náročné metody.[3]

4.1.1 Interpolace nejbližším sousedem

Nejméně náročným způsobem snížení rozlišení snímku je interpolace nejbližším sousedem, kdy se jasová hodnota pixelu po změně měřítka rovná jasové hodnotě nejbližšího pixelu před změnou měřítka. Snižování velikosti obrazu touto metodou na $1/n$ původní velikosti (kde n je celé číslo) bývá označováno jako decimace. Při decimaci je ze snímku zachován pouze každý n-tý pixel v obou rozměrech. Pro zamezení vzniku artefaktů je před decimací vhodné zredukovat šum například konvolucí Gaussovým filtrem.[3]

4.1.2 Bilineární interpolace

Při bilineární interpolaci je nová jasová hodnota změně měřítka pixelu určena na základě čtyř nejbližších pixelů před změnou měřítka.[3]

$$\begin{aligned} I(p) = &\frac{I(q_{00})}{(x_1 - x_0)(y_1 - y_0)}(x_1 - x)(y_1 - y) \\ &+ \frac{I(q_{10})}{(x_1 - x_0)(y_1 - y_0)}(x - x_0)(y_1 - y) \\ &+ \frac{I(q_{01})}{(x_1 - x_0)(y_1 - y_0)}(x_1 - x)(y - y_0) \\ &+ \frac{I(q_{11})}{(x_1 - x_0)(y_1 - y_0)}(x - x_0)(y - y_0) \end{aligned} \tag{4.1}$$

Kde:

$I(p)$ je jasová hodnota nového pixelu na souřadnicích x, y

$I(q_{ab})$ je jasová hodnota pixelu na souřadnicích x_a, y_b

Tento vztah se pro zmenšování měřítka na $1/n$ původní velikosti kde n je sudé číslo zjednoduší tak, že jasová hodnota nového pixelu je aritmetickým průměrem jasových hodnot čtyř nejbližších pixelů.[3]

4.2 Konvoluce

Konvoluce je jednou ze základních matematických operací využitelných pro zpracování signálů, pro dvojrozměrný signál (v případě této práce obraz) je dána následujícím vztahem.[4]

$$g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x-dx, y-dy) \tag{4.2}$$

Kde $g(x,y)$ je výsledek konvoluce signálu $f(x,y)$ jádrem ω

Pomocí konvoluce Gaussovým filtrem $w(x,y)$ lze dosáhnout rozmazání obrazu a redukce šumu.[4]

$$w(x,y) = \frac{1}{2\pi^2} e^{\frac{-(x^2+y^2)}{\sigma^2}} \tag{4.3}$$

4.3 Geometrické transformace

Pro odstranění vlivů optické soustavy je nutné obraz upravit pomocí geometrických transformací. Velmi běžné je například radiální zkreslení dané tvarem objektivu, které lze popsat pomocí následujícího vztahu.[5]

$$\begin{aligned} x' &= x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \\ y' &= y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \end{aligned} \tag{4.4}$$

Kde:

x' a y' jsou souřadnice pixelu před kompenzací zkreslení

x a y jsou souřadnice pixelu po kompenzaci zkreslení

k_1 až k_3 jsou korekční koeficienty

r je euklidovská vzdálenost pixelu $p(x,y)$ od středu zkreslení (typicky je střed zkreslení stejný jako střed obrazu)

Dalším běžným typem zkreslení je tangenciální zkreslení. To lze popsat pomocí následujícího vztahu.[5]

$$\begin{aligned} x' &= x + 2\rho_1 \cdot y + \rho_2 \cdot \left(r^2 + 2x^2\right) \\ y' &= y + 2\rho_2 \cdot x + \rho_1 \cdot \left(r^2 + 2y^2\right) \end{aligned} \tag{4.5}$$

Kde:

x' a y' jsou souřadnice pixelu před kompenzací zkreslení

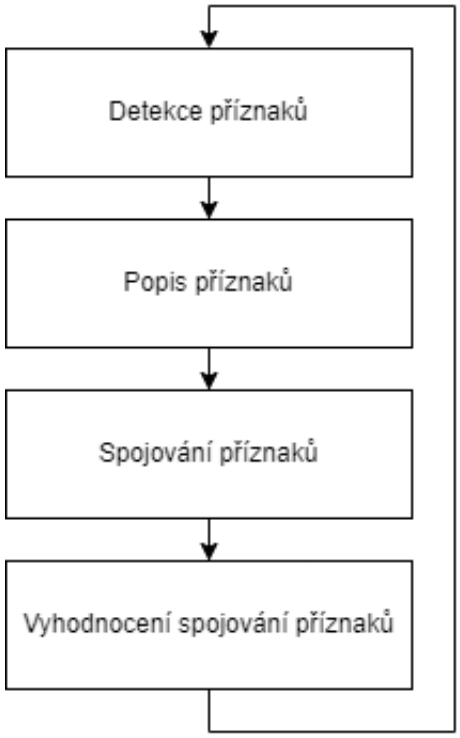
x a y jsou souřadnice pixelu po kompenzaci zkreslení

ρ_1 a ρ_2 jsou korekční koeficienty

r je euklidovská vzdálenost pixelu $p(x,y)$ od středu zkreslení

5 Zpracování obrazových dat

Pro určení polohy bezpilotního letadla zvolenou metodou jsou nutné určité referenční body. V rámci této práce jsou referenční body určovány na základě obrazových dat následujícím postupem:



Obr. 5.1: Postup zpracování obrazových dat.

5.1 Detekce příznaků

V první fázi zpracování dat je nutné detekovat v obrazu příznaky vhodné pro další sledování. Příznakem mohou typicky být myšleny klíčové body, hrany, nebo oblasti. Vzhledem k výpočetní jednoduchosti a odolnosti vůči zkreslení daném změnou úhlu, pod kterým jsou snímány, jsou v této práci jako příznaky využity klíčové body. Klíčové body jsou definované svým umístněním na snímku, některé algoritmy ke každému rysu bodu připisují i velikost a orientaci. Prioritou pro výběr algoritmu pro detekci příznaků je v tomto případě nízká výpočetní náročnost.[6][7]

5.1.1 FAST

Jedním z nejrychlejších v praxi používaných detektorů příznaků je FAST. [6] Tento algoritmus je velmi jednoduchý:

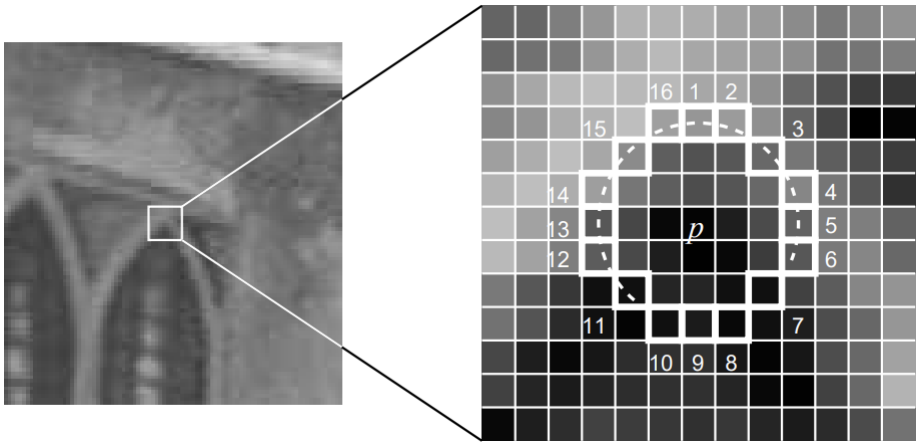
Je určena kružnice o obvodu n se středem v posuzovaném pixelu p o intenzitě I_p. Typicky se využívá $n = 16$ nebo $n = 12$.[8]

Posuzovaný pixel p je považován za platný příznak pokud každý pixel r o intenzitě I_r ležící na této kružnici splňuje podmínku:[8]

$$|I_p - I_r| > t \tag{5.1}$$

Kde:

t je hranice detekce.



Obr. 5.2: Ilustrace příznaku detekovaného algoritmem FAST.[8]

Tento algoritmus má řadu nevýhod, nejvýraznějšími z nich je velmi nízká odolnost vůči šumu, rozmazání obrazu a změně měřítka. V praxi tato metoda produkuje velké množství nespolehlivých příznaků, které se při velmi malých změnách obrazu „ztrácí“ a znovu „objevují“. Tento problém je umocněn tím, že samotný algoritmus FAST neurčuje ukazatele kvality příznaku. Nespolehlivé příznaky tedy nelze bez dalšího zpracování odlišit od více spolehlivých.[8]

5.1.2 Moravcův detektor

Jednou z nejstarších metod pro detekci bodových příznaků je Moravcův detektor. Pro nalezení příznaků v obrazu lze využít následující vztah:[9]

$$E(x,y) = \sum_{u,v} w(u,v)[I(x+u, y+v) - I(x,y)]^2 \tag{5.2}$$

Kde:

E popisuje „kvalitu“ příznaků v posuzovaném pixelu
x a y jsou pixelové souřadnice bodu v obrazu
w je obdélníková okénková funkce
I je intenzita daného pixelu
u a v určují posun oproti posuzovanému pixelu

Detekované příznaky jsou pak lokální maxima $E(x,y)$

Moravcův detektor je velmi jednoduchý, má však několik nedostatků. Mezi ty patří zejména velké změny v určené kvalitě příznaku v závislosti na změně orientace snímaného bodu a nízká odolnost vůči šumu. Další značnou nevýhodou je to, že algoritmus nezohledňuje symetrii okolí bodu, detekuje proto i příznaky ležící na hranách, které nelze spolehlivě přesně sledovat v prostoru.[9]

5.1.3 Harrisův detektor

Harrisův detektor je založený na Moravcově detektoru. Využívá stejnou základní koncepci s několika úpravami. Moravcův detektor využívá následující vztah:[9]

$$E(x,y) = \sum_{u,v} w(u,v)[(x+u, y+v) - I(x,y)]^2 \tag{5.2}$$

První úpravou je změna okénkové funkce w. Místo obdélníkové funkce využívá Harrisův detektor Gaussův filtr:[9]

$$w(u,v) = e^{\frac{-(u^2+v^2)}{\sigma^2}} \tag{5.3}$$

Moravcův detektor určuje změnu intenzity pomocí posunu o celočíselný počet pixelů u,v. Změna intenzity je tedy určována pouze v několika diskrétních směrech.[9]

$$[I(x+u, y+v) - I(x,y)]$$

Například pro typické $u, v \in \{-1; 1\}$ je změna intenzity počítána pro 9 pixelů obklopujících posuzovaný bod, je tedy určována pouze ve směrech vzájemně posunutých o 45°. To způsobuje nekonzistentní výsledky v závislosti na otočení obrazu.[9]

Harrisův detektor využívá k vyřešení tohoto problému aproximaci výpočtu změny intenzity pomocí Taylorovy řady:[9]

$$[I(x+u, y+v) - I(x,y)] \approx [u \cdot I_x + v \cdot I_y] = \begin{bmatrix} u & v \end{bmatrix} \cdot \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}$$

Kde:

I_x a I_y jsou derivace intenzity v okolí posuzovaného pixelu v daných směrech.

Celkový výpočet kvality příznaku je tedy:[9]

$$E(x,y) = \sum_{u,v} w(u,v) \cdot \begin{bmatrix} u & v \end{bmatrix} \cdot \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}^2 \tag{5.4}$$

Kde:

E popisuje „kvalitu“ příznaků v posuzovaném pixelu

I_x a I_y jsou derivace intenzity v okolí posuzovaného pixelu v daných směrech

u a v určují posun oproti posuzovanému pixelu

w je okénková funkce

Stejně jako u Moravcova detektoru jsou detekované příznaky lokální maxima $E(x,y)$[9]

Další výhodou Harrisova detektoru je možnost určit, jestli je detekovaný příznak „roh“ nebo „hrana“.[9]

$$M = \sum_{u,v} w(u,v) \cdot \begin{bmatrix} I_x^2 & I_x \cdot I_y \\ I_x \cdot I_y & I_y^2 \end{bmatrix}^2 \tag{5.5}$$

Hranu by značil velký rozdíl mezi hodnotami I_x a I_y. Pro roh jsou hodnoty I_x a I_y podobné. Hrany a rohy lze rozlišit pomocí parametru R:[9]

$$R = det(M) - k \cdot stopa(m)^2 \tag{5.6}$$

Kde:

R určuje symetrii příznaku

k je koeficient, typicky se využívá hodnota 0.05 [9]

$stopa(m)$ je suma prvků na hlavní diagonále M

Vzhledem k tomu, že hrany nejsou vhodné pro přesné sledování v prostoru, se z množiny sledovaných příznaků obvykle vyřazují příznaky s příliš nízkým parametrem R.[9]

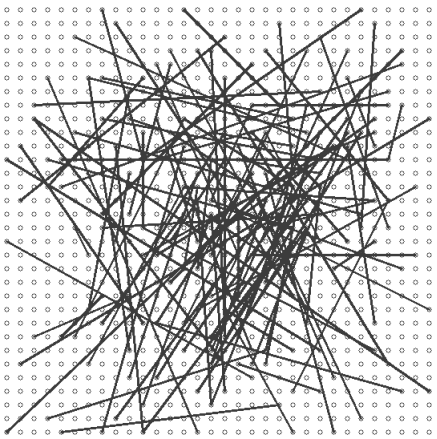
5.2 Popis příznaků

Za účelem spojování příznaků lze ke každému příznaku určit tzv. deskriptor, který popisuje okolí příznaku. Algoritmy pro popis příznaků lze posuzovat podle jejich

rychlosti a odolnosti vůči změně jasu a měřítka, rotaci a šumu. Vzhledem k tomu, že jsou deskriptory využívány pro spojování příznaků z téměř identických snímků - snímků pořízených ve stejnou dobu stereokamerou, nebo dvou po sobě následujících snímků ze stejného snímače - není nutná vysoká odolnost vůči výše zmíněným vlivům. Na druhou stranu je velmi důležitá rychlost algoritmu a jednoduchost dalšího zpracování.[7]

5.2.1 BRIEF

Algoritmus BRIEF je jedním z nejjednodušších a nejrychlejších prakticky využívaných deskriptorů.[6] Před výpočtem deskriptoru může být obraz pro zvýšení odolnost vůči šumu a malým změnám obrazu předzpracován konvolucí Gaussovým filtrem. Dále jsou vybrány dvojice pixelů pro výpočet deskriptoru. Aby bylo možné deskriptory vzájemně porovnávat a spojovat, je rozmístění těchto dvojic neměnné.[10]



Obr. 5.3: Rozmístění dvojic pixelů generované na základě Gaussova rozdělení.[10]

Pro každou dvojici pixelů a a b z celkového počtu dvojic n je provedeno porovnání podle intenzity, pravdivostní hodnota tohoto porovnání pak udává bit na k-té pozici deskriptoru[10] :

$$D(k) = \begin{cases} 1 & pokud \quad I_a(k) > I_b(k) \\ 0 & pokud \quad I_a(k) \leq I_b(k) \end{cases} \tag{5.7}$$

Kde:

D je deskriptor zapsaný jako bitový vektor

I_a a I_b jsou intenzity dvojice porovnávaných pixelů

k je index bitu deskriptoru korespondující s indexem dvojice porovnávaných pixelů, nabývá hodnot $\langle 0; n-1 \rangle$ kde n je délka deskriptoru v bitech

Vzhledem k tomu, že deskriptor BRIEF je bitový vektor, se určuje odlišnost mezi dvěma deskriptory jako jako jejich Hammingova vzdálenost:[10]

$$R_{1,2} = \sum_{k=0}^{n-1} D_1(n) \oplus D_2(n) \tag{5.8}$$

Kde:

D_1,D_2 jsou porovnávané deskriptory

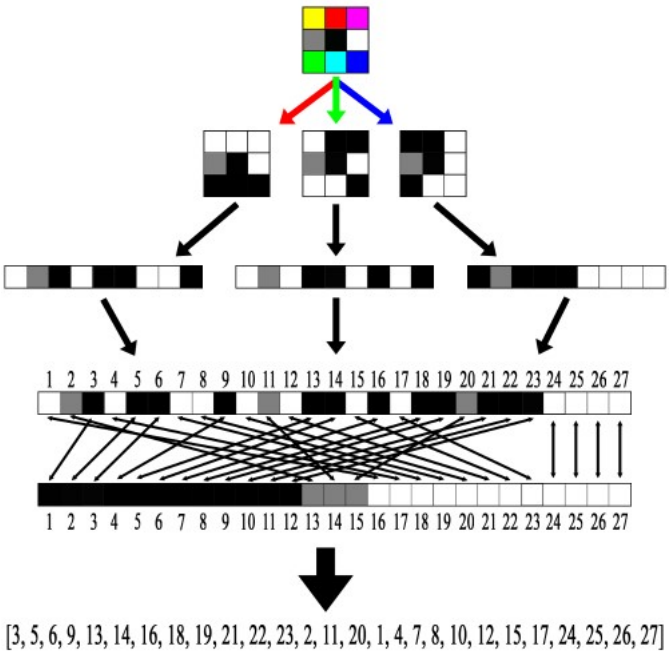
$R_{1,2}$ je odlišnost porovnávaných deskriptorů

5.2.2 LUCID

Deskriptor LUCID představuje rychlejší alternativu k deskriptoru BRIEF. Algoritmus jde rozdělit na tyto základní kroky:

- Rozdělení popisované oblasti na barevné složky.
- Vektorizace jasových hodnot pixelů barevných složek.
- Konkatenace vektorů, přiřazení indexu ke každé hodnotě.
- Seřazení prvků konkatenovaného vektoru podle jasu.

Deskriptorem je pak vektor indexů přiřazených ve třetím kroku v pořadí, ve kterém jsou po seřazení v posledním kroku.[11]



Obr. 5.4: Ilustrace jednotlivých kroků výpočtu deskriptoru LUCID. [11]

Nevýhodou tohoto deskriptoru je vysoká citlivost na šum, rotaci a změnu měřítka obrazu. Výhodou je možnost zohlednění barvy v tvorbě deskriptoru. Pro další zrychlení výpočtu lze místo barevných složek využít pouze jasové hodnoty pixelů.

Odlišnost dvou deskriptorů je obdobně jako pro BRIEF určena jako jejich Hammingova vzdálenost.[11]

5.3 Spojování příznaků

Pro sledování příznaků je nutné vzájemně spárovat příznaky z obrazu z obou kamer a pro minimálně jednu kameru spojit příznaky z nejnovějšího snímku s příznaky z minulých snímků. Vzhledem k tomu, že jsou v této práci využívány pouze binární deskriptory, je pro porovnávaní prováděno výpočtem odlišnosti (sekce 5.8) pro každou možnou dvojici deskriptorů. Výhodou je, že tento přístup zaručuje nalezení nejlepšího páru deskriptorů, tedy páru deskriptorů s nejnižší odlišností a konzistentní výpočetní náročnost. Nevýhodou je relativně vysoká výpočetní náročnost:[14]

$$K_n(n) = n^2 \tag{5.9}$$

Kde:
K_n je nutný počet porovnání
n je počet porovnávaných deskriptorů

5.4 Metody upravující spojování příznaků

V rámci této práce jsou příznaky spojovány na snímcích s malým časovým a prostorovým odstupem, toho lze využít následujícími způsoby:

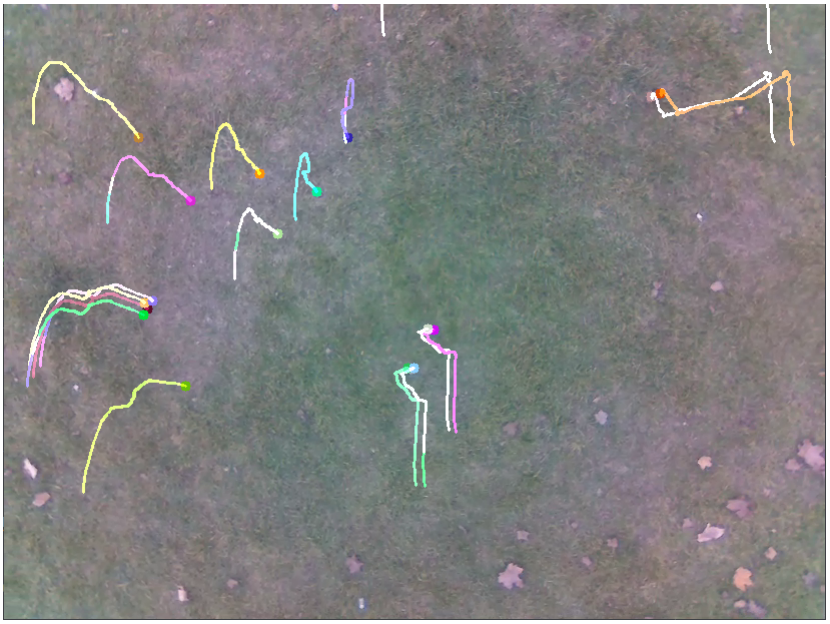
5.4.1 Odhad polohy příznaku pomocí optického toku

Optický tok udává rozložení zdánlivých rychlostí pohybu objektů na snímku. Data z optického toku lze využít pro odhad změny polohy příznaků. Při určování nové polohy příznaků tedy stačí prohledat okolí odhadované polohy, nikoliv celý snímek. Nevýhodou je riziko selhání způsobené špatným odhadem pohybu rysu. To může být způsobené například velkými rozdíly zdánlivých rychlostí v okolí rysu. Pro výpočet optického toku jsem kvůli nízké výpočetní náročnosti využil algoritmus Lucas-Kanade, který je pro dvojrozměrný obraz daný následujícím vztahem[12]:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(p_i)^2 & \sum_i I_x(p_i) I_y(p_i) \\ \sum_i I_y(p_i) I_x(q_i) & \sum_i I_y(p_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(p_i) I_t(p_i) \\ -\sum_i I_y(p_i) I_t(p_i) \end{bmatrix} \tag{5.10}$$

V_x a V_y jsou zdánlivé rychlosti posuzovaného pohybu bodu ve směru os x a y
I_x a I_y jsou derivace intenzity ve směru os x a y
p je pixel v okolí posuzovaného bodu (včetně samotného posuzovaného bodu), typicky je využívána oblast 3x3 se středem v posuzovaném bodu[12]

Na obrázku 5.5 je znázorněn zdánlivý pohyb bodů určený metodou Lucas-Kanade.



Obr. 5.5: Trasa bodů sledovaných metodou Lucas-Kanade.

5.4.2 Odhad polohy využívající extrapolaci pohybu rysu

Existuje řada algoritmů sloužících k odhadu budoucí polohy rysu na základě minulých zaznamenaných poloh, jedním z nejvíce rozšířených je tzv. Kálmánův filtr. Nevýhodou těchto algoritmů je jejich nepřesnost při nedostatečném množství informací o minulých stavech, nefungují tedy pro nově detekované rysy.[13]

5.5 Vyhodnocení spojování příznaků

5.5.1 Výběr nejlepšího páru

Tato metoda je velmi jednoduchá, výstupem sledovaní příznaků jsou páry deskriptorů s nejnižší vzájemnou odlišností.[14]

5.5.2 Porovnání n nejlepších páru

Pro vyhodnocení výstupu sledování příznaků je pro každý deskriptor porovnáváno n nejlepších párů, v praxi se téměř vždy volí hodnota $n = 2$. Pro $n = 2$ lze vyhodnocení provést pomocí následujícího výrazu:[14]

$$R_{1,x}(0) > R_{1,x}(1) \cdot k \tag{5.11}$$

Kde:

$R_{1,x}(0)$ je nejnižší naměřená odlišnost mezi deskriptorem D_1 a libovolným deskriptorem D_{xa}

$R_{1,x}(1)$ je druhá nejnižší naměřená odlišnost mezi deskriptorem D_1 a libovolným deskriptorem D_{xb}

D_{xa} a D_{xv}jsou deskriptory s nejnižší a druhou nejnižší (v tomto pořadí) odlišností vzhledem k D_1.

Pokud je tento výraz pravdivý, jsou deskriptory D_1 a D_{xa}, tedy deskriptory odpovídající odlišnosti $R_{1,x}(0)$, považovány za platný pár.[14]

5.6 Vybraný algoritmus pro zpracování obrazových dat

Pro detekci a popis příznaků jsem se rozhodl využít algoritmus ORB (z angl. *Oriented FAST rotated BRIEF*), ten má následující kroky.

Nejdříve je ze snímku vytvořena N-úrovňová obrazová pyramida. Pro tuto pyramidu platí, že $k-1$ vrstva vznikne z k-té vrstvy konvolucí Gaussovým filtrem a následnou decimací obrazu. Decimací je myšleno snížení rozlišení výběrem určitého množství rovnoměrně rozmístěných pixelů. Obvykle se decimací snižuje počet pixelu na 25 % původního množství, v tomto případě jsou vybrány pixely s lichými souřadnicemi x a y v rámci obrazu. N-tá vrstva pyramidy je neupravený obraz. Využití obrazové pyramidy snižuje vliv velikosti obrazu na detekci a popis příznaků.[15]

Pomocí algoritmu FAST (popsaný v sekci 5.1.1) jsou poté na všech vrstvách pyramidy nalezeny příznaky. Z těch je potom pomocí Harrisova detektoru (popsaný v sekci5.1.3) vybrán požadovaný počet kvalitních příznaků, jsou tedy vybírány příznaky s nejvyšší hodnotou E dle rovnice 5.5. V okolí vybraných příznaků je pak vypočten moment intenzity m:[15]

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{5.12}$$

Pro tento výpočet je obraz posunutý tak, aby souřadnice středu příznaku byly $x = 0$, $y = 0$. Dále lze určit centroid intenzity C. [15]

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{5.13}$$

Pro úhel θ svíraný kladnou poloosou x a vektorem s počátkem ve středu příznaku a koncem v centroidu intenzity C platí následující vztah: [15]

$$\theta = atan2(m_{01}, m_{10}) \tag{5.14}$$

Příznaky jsou poté popsány pomocí deskriptoru BRIEF (popsaný v sekci 5.2).Tento algoritmus je upraven tak, že rozmístění dvojic pixelů určených pro výpočet deskriptoru je otočené o úhel θ. To značně snižuje závislost deskriptoru na orientaci příznaku.[15]

Deskriptory byly spojovány metodou popsanou v sekci 5.3 a spojení byla vyhodnocována pomocí metody porovnání n nejlepších páru. Jsou spojovány příznaky detekované levou a pravou kamerou. Dále jsou spojovány příznaky detekované pravou kamerou v současném a minulém snímku.

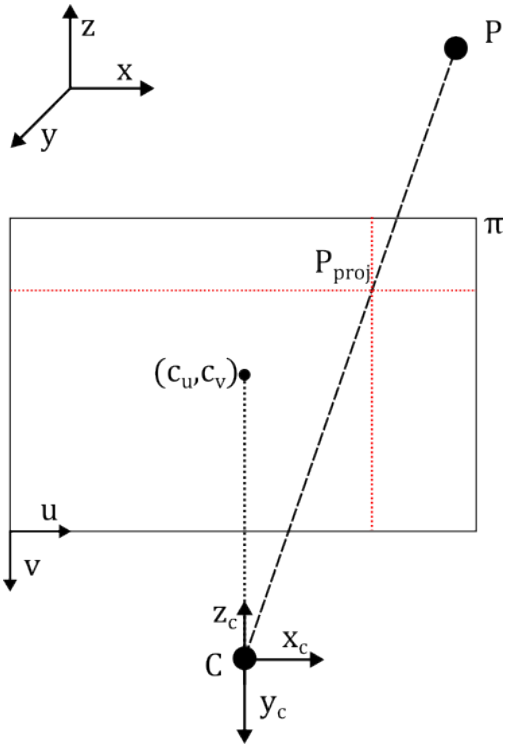
Za referenční bod jsou považovány úspěšně spojené příznaky z obrazů z levé a pravé kamery. Pokud se referenční bod nepodaří spojit s příznakem z minulého snímku je považován za nový.

Metody upravující spojování příznaků (zmíněné v kapitole 5.4) nebyly využity, protože pozorované zlepšení sledování příznaků bylo příliš malé, aby vyvážilo přidanou výpočetní náročnost a rizika selhání spojená s těmito metodami. Využití těchto metod by také značně zkomplikovalo ladění systému.

6 Zpracování referenčních bodů

6.1 Výpočet relativních souřadnic referenčního bodu

Vztah mezi polohou bodu v prostoru a obrazem zachyceným kamerou lze znázornit následujícím schématem:[17]



Obr. 6.1: Schéma modelu kamery.

Každý bod P v souřadnicové soustavě dané vztažným bodem dronu x, y, z je promítnut do roviny kamery π jako bod P_{proj}. Model kamery je daný středem projekce C, který je počátkem souřadnicové soustavy kamery x_c, y_c, z_c. Za střed souřadnicové soustavy dané vztažným bodem dronu můžeme v kontextu zpracování referenčních bodů považovat například těžiště dronu. Vztah mezi bodem P a jeho projekcí P_{proj} je následující:[17]

$$P_{proj}(u, v) = KP(x_c, y_c, z_c) \tag{6.1}$$

Kde K je matice intrinsických (vnitřních) parametrů kamery. Tento vztah lze dále upravit:[17]

$$\begin{bmatrix} u_P \\ v_P \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{F}{q_u} & 0 & c_x \\ 0 & \frac{F}{q_v} & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{cP} \\ y_{cP} \\ z_{cP} \end{bmatrix} \tag{6.2}$$

u_P, v_P jsou pixelové souřadnice bodu P_{proj}
x_{cP}, y_{cP}, z_{cP} jsou souřadnice bodu P v souřadnicové soustavě kamery
F je ohnisková vzdálenost kamery v metrech
q_u a q_v jsou rozměry pixelů ve směrech u a v v metrech
c_x a c_y jsou pixelové souřadnice kolmého průmětu středu projekce C do roviny π

Tento vztah je normalizovaný pro souřadnici z_c. Bod $P(x_c, y_c, z_c)$ lze dále ze souřadnicové soustavy kamery převést do souřadnicové soustavy dané vztažným bodem dronu:[17]

$$P(x_c, y_c, z_c) = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} P(x, y, z) \tag{6.3}$$

Kde R je matice popisující vzájemnou rotaci souřadnicových soustav, t je vektor popisující vzájemný posun jejich počátků. Platí tedy:[17]

$$\begin{bmatrix} x_{cP} \\ y_{cP} \\ z_{cP} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_P \\ y_P \\ z_P \\ 1 \end{bmatrix} \tag{6.4}$$

x_{cP}, y_{cP}, z_{cP} jsou souřadnice bodu P v souřadnicové soustavě kamery
r je parametr popisující rotaci
t je parametr popisující posun v dané ose

Tyto vztahy lze zapsat v následujícím tvaru:

$$P_{proj}(u, v) = K \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} P(x, y, z) \tag{6.5}$$

Souřadnice bodu P (x, y, z) lze potom jednoznačně určit na základě výpočtu dle předchozího vztahu pro dva snímky vyfocené z různých pozic:[17]

$$\begin{aligned} P_{proj}(u_1, v_1) &= K_1 \begin{bmatrix} R_1 & t_1 \\ 0 & 1 \end{bmatrix} P(x, y, z) \\ &\wedge \\ P_{proj}(u_2, v_2) &= K_2 \begin{bmatrix} R_2 & t_2 \\ 0 & 1 \end{bmatrix} P(x, y, z) \end{aligned} \tag{6.6}$$

6.1.1 Přepočet souřadnic bodů

Prvním krokem je výpočet polohy každého referenčního bodu $P_{(n)}$, kde n nabývá hodnot od 1 do celkového počtu referenčních bodů, vzhledem ke vztažnému bodu dronu podle vztahů uvedených v kapitole 6.1.

Poté jsou tyto souřadnice na základě dat z akcelerometru a kompasu bezpilotního letadla převedeny do souřadnicové soustavy s počátkem v těžišti bezpilotního letadla T, kde kladný směr X směřuje k magnetickému severu, kladný směr Y směřuje k magnetickému východu a kladný směr Z proti směru působení gravitace.

$$P_{(n)}(x, y, z) \longrightarrow P_{(n)}(X, Y, Z) \tag{6.7}$$

Dále je určena skutečná poloha těžiště bezpilotního letadla. Tu lze určit součtem absolutní polohy známého referenčního bodu P, $P_{(n)}(X_s, Y_s, Z_s)$ a polohy referenčního bodu P vzhledem k těžišti bezpilotního letadla T $P_{(n)}(X, Y, Z)$

$$T_{(n)} \begin{cases} X_s = P_{(n)}(X_s) + P_n(X) & [m] \\ Y_s = P_{(n)}(Y_s) + P_n(Y) & [m] \\ Z_s = P_{(n)}(Z_s) + P_n(Z) & [m] \end{cases} \tag{6.8}$$

Za skutečnou polohu T je považován medián všech vypočtených $T_{(n)}$

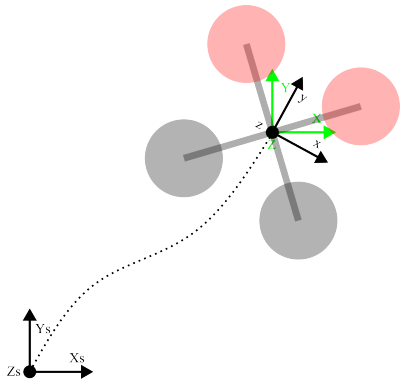
Dále je obdobným způsobem určena absolutní poloha všech nově detekovaných referenčních bodů $P_{(n)}$

$$P_{(n)} \begin{cases} X_s = P_{(n)}(X) - T(X_s) & [m] \\ Y_s = P_{(n)}(Y) - T(Y_s) & [m] \\ Z_s = P_{(n)}(Z) - T(Z_s) & [m] \end{cases} \tag{6.9}$$

Při inicializaci je absolutní poloha bodu T nastavena na 0

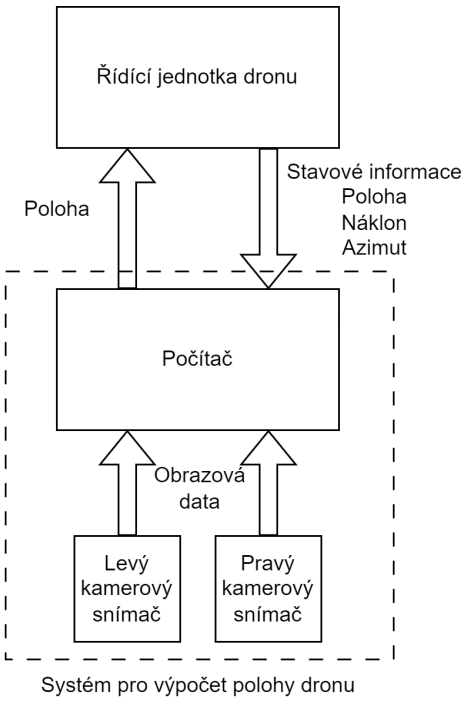
$$T_{(n)} \begin{cases} X_s = 0 \\ Y_s = 0 \\ Z_s = 0 \end{cases} \tag{6.10}$$

Na obrázku 6.2 je znázorněný přehled využívaných souřadnicových systémů



Obr. 6.2: Přehled využívaných souřadnic.

7 Výběr hardwaru



Obr. 7.1: Komunikační schéma systému.

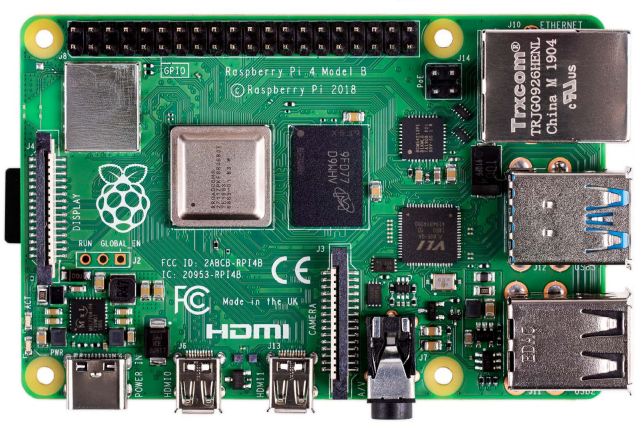
Systém pro výpočet polohy dronu se skládá z počítače a dvou kamerových snímačů. Je navržený pro použití s libovolnou řídicí jednotkou dronu podporující komunikační protokol MAVLink. Pro otestování implementace byla využita řídicí jednotka Pixhawk Hex Cube Black. Kamerové snímače jsou k počítači připojené pomocí běžně využívaného rozhraní CSI. Systém tedy není závislý na využití specifického modelu kamery. Na obrázku 7.1 je znázorněno komunikační schéma systému.

7.1 Počítač

7.1.1 Raspberry Pi

První testy jsem provedl na jednodeskovém počítači Raspberry Pi 4B, 4GB RAM. Tento počítač jsem vybral pro jeho rozšířenost a tedy dobrou obecnou softwarovou podporu. Využívá čtyřjádrový procesor ARM Cortex A-72. [18]

Nevýhodou je absence dedikované grafické procesorové jednotky, kvůli které musela být obrazová data zpracovávána pro tuto činnost neefektivním procesorem. Kvůli tomuto nedostatku jsem od využití Raspberry Pi upustil.



Obr. 7.2: Raspberry Pi 4B.[19]

7.1.2 Jetson Nano

Převážná většina testů byla provedena na počítači Jetson Nano. Ten je založený na procesoru ARM Cortex A-57 - předchůdci procesoru využitého v Raspberry Pi 4B. Hlavní výhodou tohoto počítače je grafická procesorová obsahující 128 CUDA jader. [20]

To umožňuje efektivní zpracovaní obrazových dat. Platforma CUDA je také jedna z nejrozšířenějších a má proto všeobecně dobrou softwarovou podporu. Tento počítač má dostatečný výpočetní výkon pro zpracování a vyhodnocování obrazu, zároveň má přijatelně nízkou hmotnost, spotřebu energie i cenu. Je proto využit v testech popsaných v této práci i v implementaci projektu.

Obr. 7.3: Jetson Nano Developer Kit.[21]

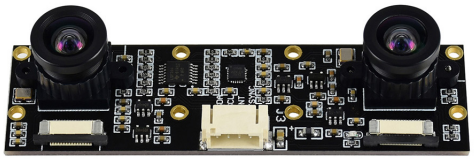
7.2 Kamera

7.2.1 Stereokamera IMX219-83

Pro testování jsem využil stereokameru IMX219-83. Tato kamera má dva snímače o rozlišeni 8 Mpx umístěné na jedné deska plošných spojů se vzájemnou vzdáleností 6 cm. Každý snímač je schopný zaznamenávat video o rozlišení až 3280 na 2464 pixelů.[22]

Umístění obou kamer do jednoho modulu zjednodušuje testování, ale nízká vzdálenost mezi snímači výrazně snižuje přesnost měření vzdálenosti referenčního bodu od kamery. Z tohoto důvodu by pro praktické využití systému bylo vhodné použít stereokameru (nebo dvě oddělené kamery) s vyšší vzájemnou vzdáleností snímačů.

Další nevýhodou této kamery je nedokonalá synchronizace snímků z obou snímačů. Ta má za následek zhoršení přesnosti celého systému, vliv tohoto nedostatku stoupá s rostoucí rychlostí.



Obr. 7.4: Stereokamera IMX219-83.[22]

8 Výběr softwaru

U obou počítačů jsem pro testování využíval operační systém doporučený výrobcem. Pro Raspberry Pi to byl Raspberry Pi OS, pro Jetson Nano Linux4Tegra. V principu není vybraný a navržený software závislý na operačním systému, pro jednoduchost je doporučován Linux.

8.1 Knihovny, frameworky

8.1.1 OpenCV

OpenCV je open source knihovna zaměřená na počítačové vidění. Byla vybrána protože obsahuje implementace většiny algoritmů zpracování obrazu uvedených v této práci a podporuje hardwarovou akceleraci GPU CUDA. Mezi výhody této knihovny také patří podpora programovacích jazyků Python a C++, zvláště API pro C++ je pak velmi dobře optimalizované. [23]

8.1.2 MavSDK

MavSDK je knihovna určená pro komunikaci s řídicí jednotkou dronu pomocí protokolu MAVLink. Podporuje programovací jazyky C++ a Python. [24]

8.1.3 Jetson-utils

Tato knihovna obsahuje nástroje pro obsluhu některých specifických funkcí počítače Jetson Nano. Poskytuje například nástroje pro práci se sdílenou pamětí CPU a GPU a nástroje pro načítání obrazových dat z kamery.

8.2 Komunikace s řídicí jedotkou

Pro získávání dat z řídicí jednotky dronu byl vybrán protokol Mavlink. Tento protokol umožňuje přenos informací, příkazů a nastavování parametru. Využívá dva způsoby přenosu informací - point-to-point pro příkazy a požadavky, publish-subscribe pro datové toky.[26]

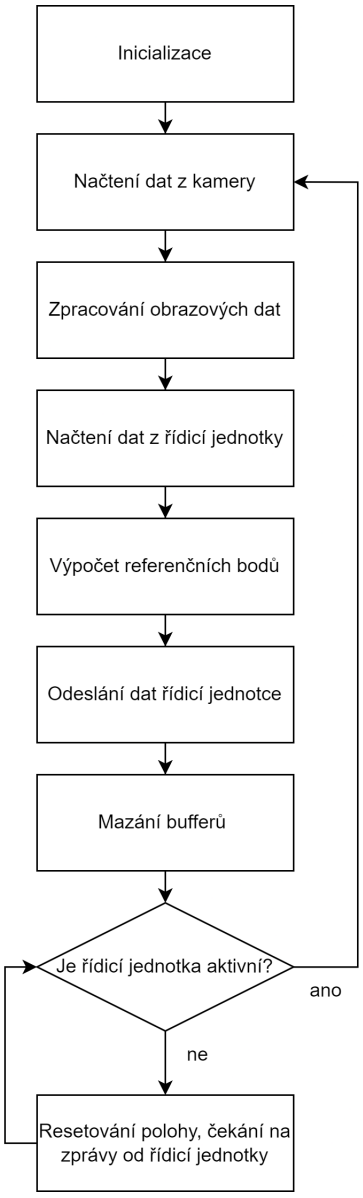
V tomto systému se využívá pro získání datového toku informací o náklonu a azimutu pro výpočet polohy, také slouží k obousměrnému přenosu dat o poloze.

Dále lze pomocí čtení stavových informací inicializovat sledování polohy až po startu. Díky tomu může být ušetřena energie a zaručené správné určení počátku souřadnicového systému.

9 Implementace

Pro jednoduchost využití je zdrojový kód implementace rozdělený do dvou zdrojových souborů jazyku C++. Soubor main.cpp obsahující vlastní zdrojový kód a config.h sloužící k nastavení parametrů programu.

Samotný běh programu lze popsat následujicím schématem:



Obr. 9.1: Schéma implementace systému.

9.1 Inicializace

V této části je navázána komunikace se stereokamerou a řídicí jednotkou. Jsou inicializované proměnné dále využívané v programu. Pro urychlení běhu programu je v této části předalokovaná značná část paměti využité programem.

9.2 Načtení dat z kamery

Pomocí nástroje Gstreamer jsou z obou snímačů načteny snímky, ty jsou uloženy do paměti grafické karty. Snímky jsou také převedeny z tříkanálového osmibitového formátu RGB do šedotónového osmibitového jednokanálového (každý pixel je popsán celočíselnou jasovou hodnotou v rozsahu 0-255).

9.3 Zpracování obrazových dat

Na oba snímky načtené v předchozím kroku je aplikován algoritmus ORB (5.6). Výstupem této operace je vektor souřadnic příznaků a matice deskriptorů těchto příznaků. Deskriptory jsou dále spojovány (5.3) - ke každému deskriptoru ze snímku z levého snímače je vybrán metodou výběru nejlepšího páru (5.5.1) deskriptor ze současného snímku z pravého snímače a deskriptor z minulého snímku levého snímače. Z výběru jsou diskvalifikovány páry deskriptorů s příliš vysokou vzájemnou odlišností a páry deskriptorů, ve kterých je jeden z deskriptorů členem jiného páru s nižší vzájemnou odlišností. Nemůže tedy nastat situace, kdy je jeden bod na jednom snímku „spojený“ s více body na druhém snímku.

Tyto operace probíhají na grafické procesorové jednotce počítače.

9.4 Načtení dat z řídicí jednotky

V tomto kroku jsou z řídicí jednotky pomocí protokolu MAVLink načtena data o poloze, náklonu a azimutu dronu. Data o náklonu a azimutu jsou nutná pro správné určení polohy referenčních bodů. Poloha je načítána proto, že řídicí jednotka dronu může pomocí jiných senzorů a filtrů zpřesnit data o poloze, které jsou tímto systémem poskytovaná. V současné verzi implementace je možné využívat pro určení výšky letu data z barometru. Využití informací o výšce letu lze zapnout v nastavení systému. Tento systém svou funkcí imituje GNSS přijímač, poskytuje tedy řídicí jednotce nefiltrovaná data o poloze, která je typicky vhodné dále zpracovat.

9.5 Výpočet referenčních bodů

Na souřadnice příznaků jsou pro odstranění zkreslení aplikované geometrické transformace popsané v sekci 4.3. Poté je vypočtena poloha postupem uvedeným v kapitole 6. Za vztažný bod dronu je považován střed levého snímače stereokamery. V případě že není dlouhodobě možné z řídicí jednotky získat informace o orientaci a azimutu předpokládá se, že jsou souřadnicové systémy x, y, z a X, Y, Z identické, tedy že kamera směřuje svisle dolů a azimut dronu je nulový.

Tyto operace probíhají na procesoru počítače.

9.6 Odeslání dat řídicí jednotce

Vypočítaná poloha je odeslána řídicí jednotce. Je využito MAVLink zprávy „VISION_POSITION _ESTIMATE“[26], pomocí které jsou odeslány souřadnice Xs, Ys, Zs vztažného bodu dronu v metrech.

10 Testování systému

10.1 Testování algoritmu zpracování obrazu

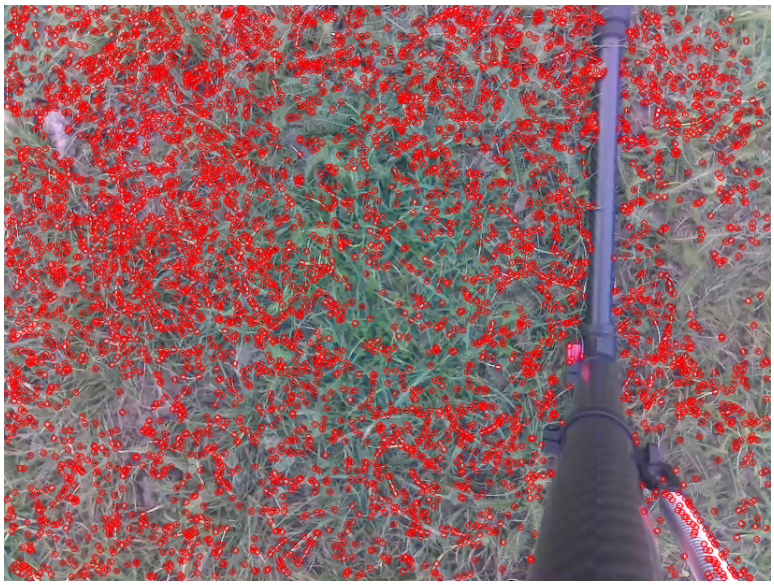
Testování proběhlo na nahraných videích ze dvou desetiminutových letů bezpilotního letadla DJI MATRICE 600 PRO. Videa byla zaznamenána pomocí stereokamery IMX219-83, kamera byla upevněná k podvozku bezpilotního letadla bez použití gimbalu a namířená směrem dolů. Video bylo ukládáno v rozlišení 820 x 616 pixelů pro každý snímač, snímková frekvence videí je 30 Hz.

Cílem testování bylo ověřit možnosti vybraného hardwaru na algoritmu zpracování obrazu. Aby byla data získaná zpracováním obrazu využitelná pro určení polohy, musí být splněna tato kritéria:

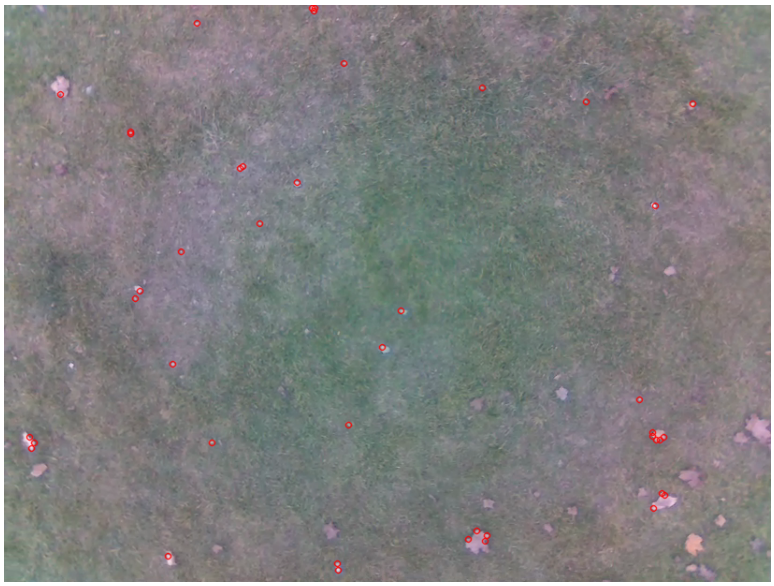
1. Referenční body musí být správně určené, referenční bod musí tedy odpovídat jednomu neměnnému místu.
2. Data musí být zpracovávána v reálném čase.
3. Vždy musí být přítomný minimálně jeden referenční bod se známou polohou, tedy referenční bod, který byl detekovaný v již předchozím snímku (kromě inicializace programu)

Úspěšného zpracování obrazových dat se mi nepodařilo docílit na počítači Raspberry Pi 4B, ale výkonnější Jetson Nano byl pro tento účel dostačující. Na obrázku 10.1 jsou znázorněny příznaky detekované algoritmem ORB.

Obr. 10.1: Příznaky detekované algoritmem ORB v testovacím videu.



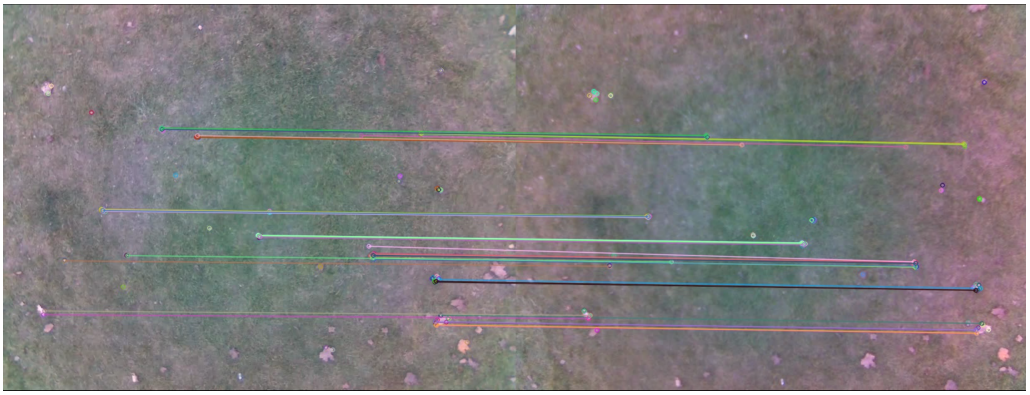
Obr. 10.2: Příznaky detekované algoritmem FAST před vzletem.



Obr. 10.3: Příznaky detekované algoritmem FAST za letu.

Z obrázků 10.2 a 10.3 je dobře patrná jedna z nevýhod algoritmu FAST, a to nemožnost nastavit požadované množství detekovaných příznaků. Přebytečné množství detekovaných příznaků v obrázku 10.2 způsobilo zpomalení programu, které by za letu mohlo způsobit selhání sledování příznaků. Tento problém se naštěstí projevuje pouze, když je snímaná plocha nepohyblivá a velmi blízko kamery, tedy když je dron

na zemi. Na obrázku 10.4 je znázorněno propojení příznaků ze snímků pořízených levým a pravým kamerovým snímačem.



Obr. 10.4: Úspěšné propojení příznaků z levé a pravé kamery je znázorněno úsečkou vedenou mezi nimi.

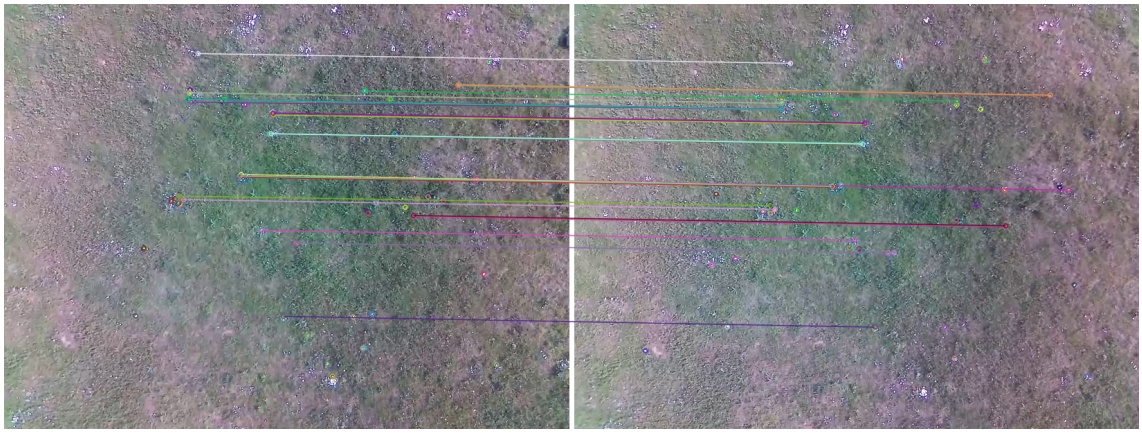
10.2 Testování implementace

Cílem těchto testů bylo ověření spolehlivosti určení polohy implementovaným systémem a určení limitací použití systému.

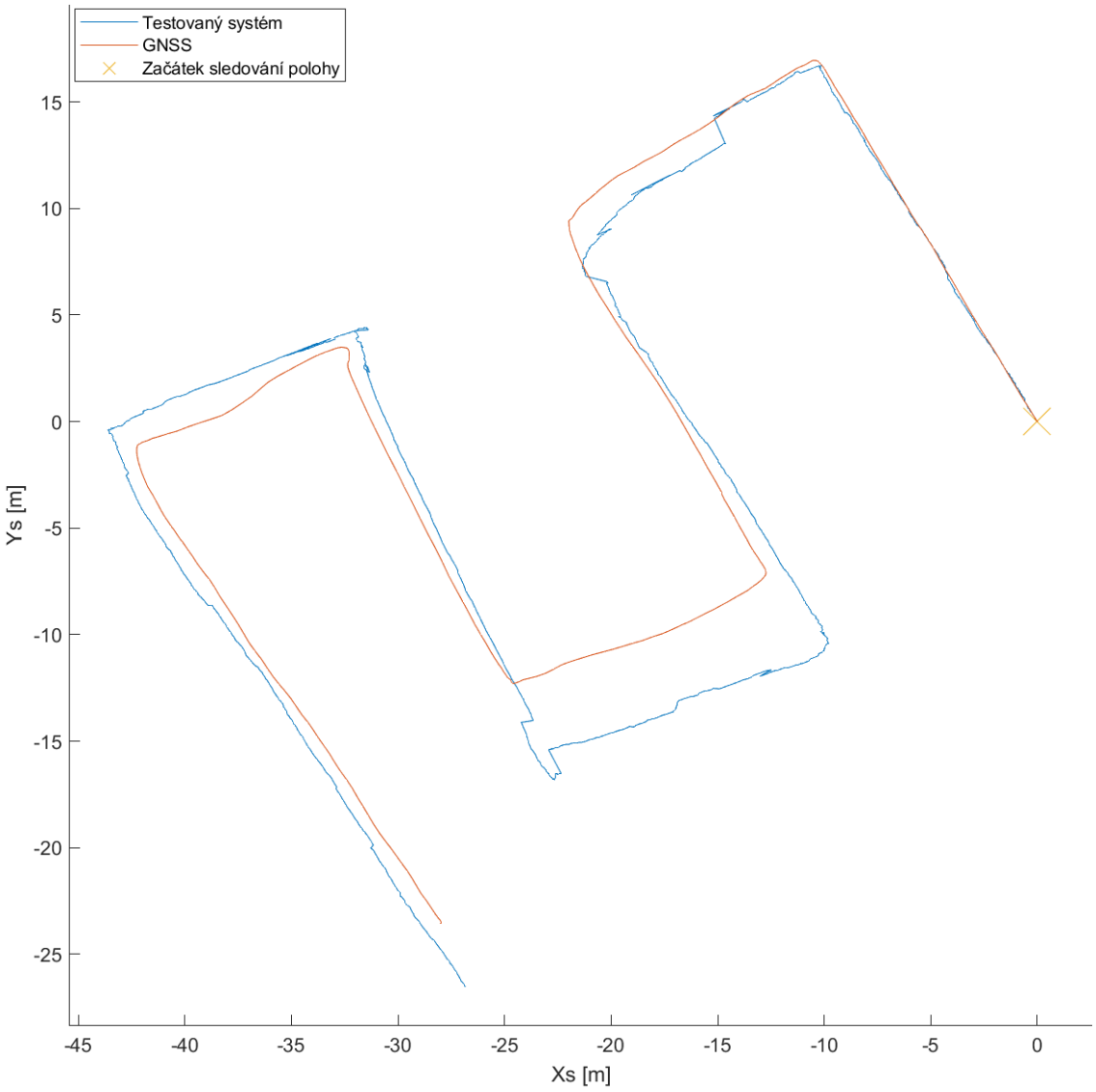
10.2.1 Testy sledování trajektorie

Následující testy byly provedeny za účelem testování schopnosti systému sledovat trasu pohybujícího se dronu a ladění parametrů systému. Software byl testovaný na videích zaznamenaných při testovacích letech s využitím stejného hardwaru jako při testování algoritmu zpracování obrazu (popsány v kapitole 10.1).

Videa byla zaznamenaná v rozlišení 820 na 616 pixelů pro každý snímač, snímková frekvence videí je 30 Hz, kamera byla umístěna na gimbalu a po dobu letu byla nasměrována svisle dolů. Data jsou porovnávána s výstupem GNSS přijímače HEX – Here 2, přesnost GNSS přijímače pohybovala v rozmezí 2 - 2,15m. Oba dva dále popsané lety byly uskutečněny v obdobných podmínkách nad travnatou plochou.



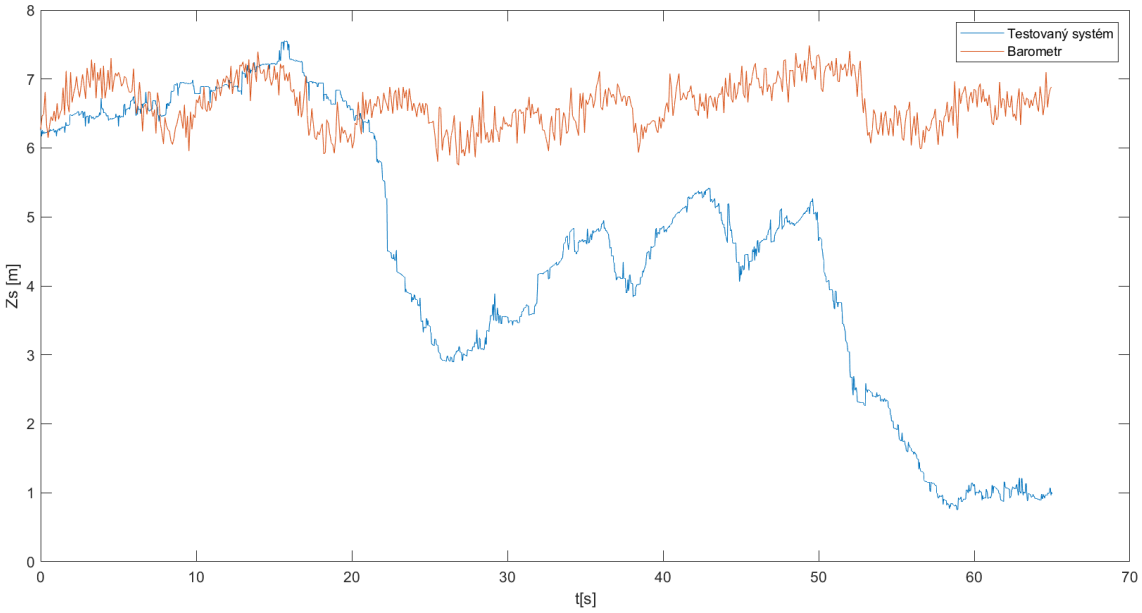
Obr. 10.5: Ukázka spojování příznaků z letu č. 1.



Obr. 10.6: Srovnání výstupu navrženého systému s výstupem GNSS přijímače, let č. 1.

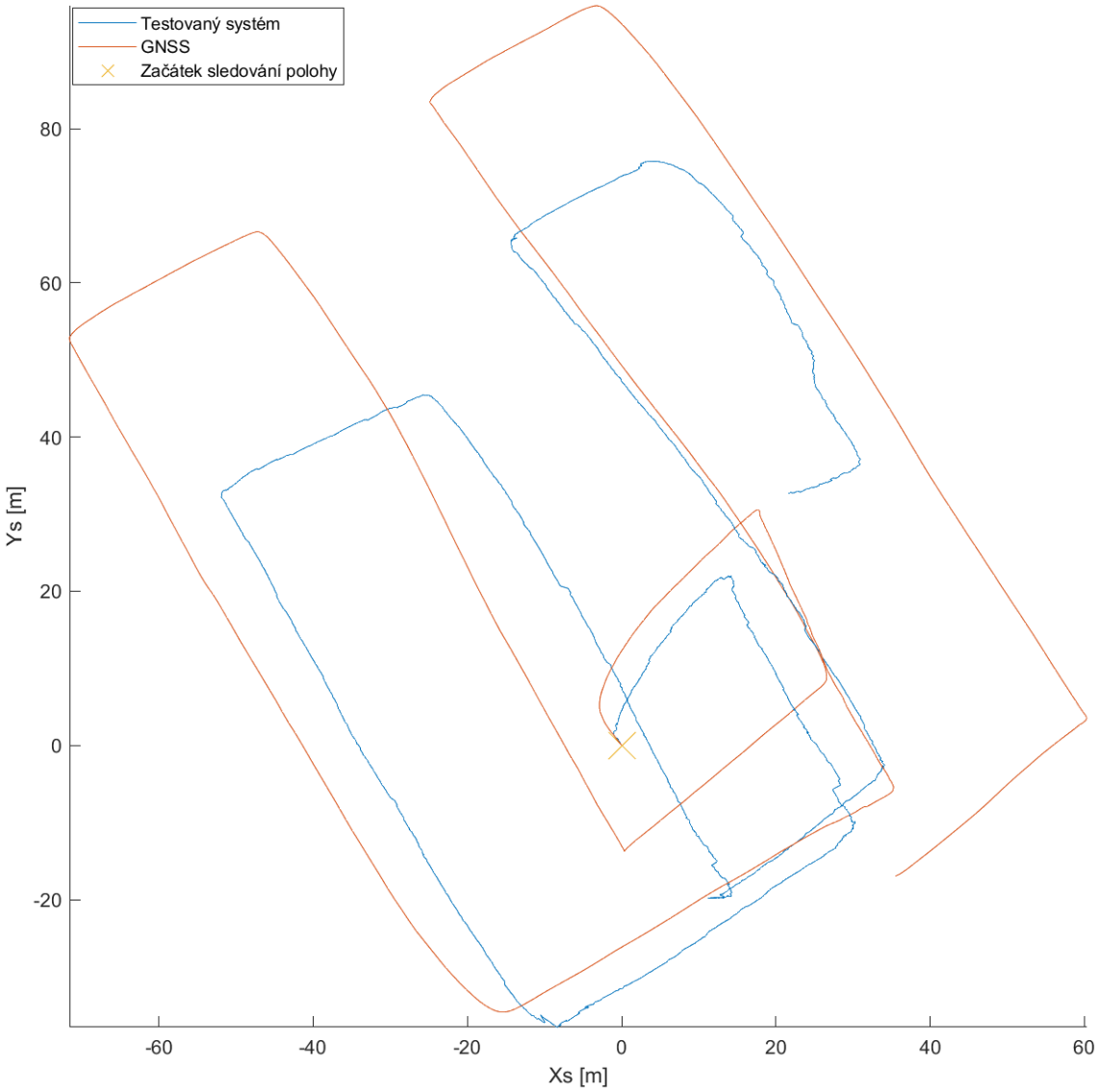
První let byl uskutečněn ve výšce přibližně 6 metrů nad terénem. Maximální odchylka v rovině dané souřadnicemi Xs a Ys byla 3,15 metru. Měřená část letu trvala 65 sekund a byla uražena celková vzdálenost přibližně 120,25 metrů. Trasa tohoto letu je znázorněna na obrázku 10.6. Na obrázku 10.5 je znázorněno spojování příznaků během tohoto letu.

Odchylka v ose Zs (pro tuto orientaci kamery koresponduje osa Zs s osou z kamery) byla výrazně vyšší než odchylka v rovině dané souřadnicemi Xs a Ys. Proto je vhodné tento systém kombinovat s jiným zdrojem informací o výšce letu, například barometrem.



Obr. 10.7: Srovnání výstupu navrženého systému s výstupem barometru MS5611, let č. 1.

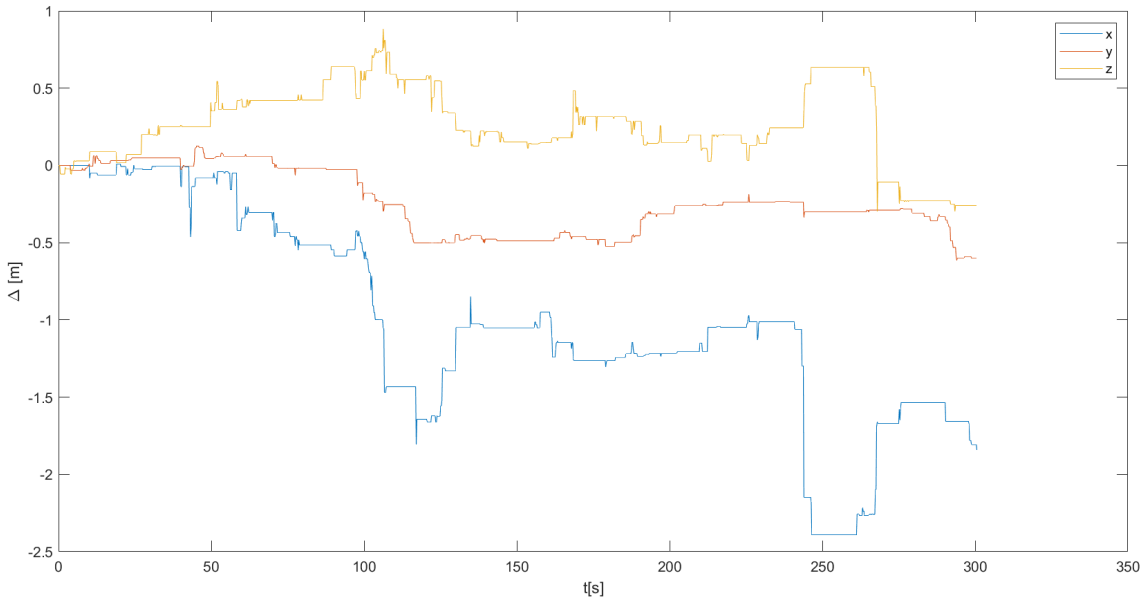
Druhý let byl proveden ve výšce přibližně 20 metrů nad terénem. Přesnost systému je v tomto případě výrazně nižší, odchylka v rovině Xs,Ys po 65 sekundách letu dosáhla 52,7 metrů. Celková uražená vzdálenost byla přibližně 676,65 metrů. Z tohoto testu je kromě absolutní odchylky rostoucí v čase patrná i multiplikativní chyba (chyba v „měřítku“). Ta je způsobená pravděpodobně nepřesným nastavením parametrů využívaných při určování polohy referenčního bodu a přepočtu souřadnicových soustav (kapitola 6). Na praktickou využitelnost systému má však relativně malý vliv protože nezkresluje informace o okamžitém směru pohybu dronu, ani o azimutu k místu startu. Schopnost stabilizace polohy a případného návratu na místo startu jsou tedy multiplikativní chybou ovlivněny pouze minimálně.



Obr. 10.8: Srovnání výstupu navrženého systému s výstupem GNSS přijímače, let č. 2.

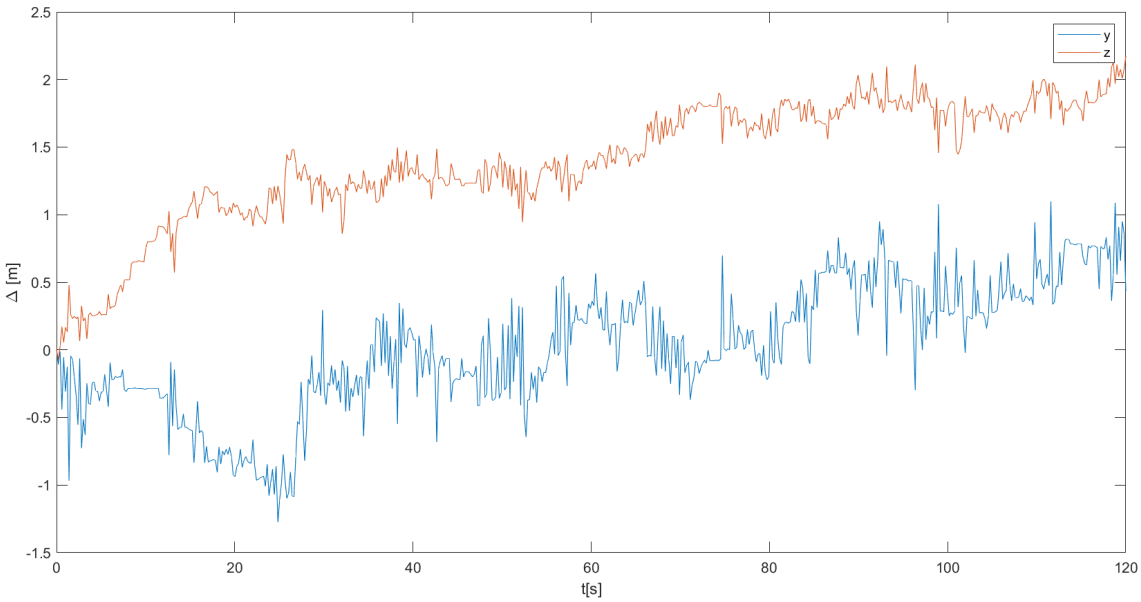
10.2.2 Testy zpracování dat v reálném čase

Tento test byl uskutečněn na „živém“ vstupu z kamery o rozlišení 1640 na 1232 pixelů na snímač. Cílem bylo ověřit schopnost systému zpracovávat obrazová data ve vysokém rozlišení v reálném čase a určit závislost absolutní chyby na čase. Při prvním testu byla změřena absolutní chyba systému při statickém umístění kamery ve výšce přibližně 4 metrů nad terénem. Sklon optické osy vzhledem k horizontu byl přibližně 45°.



Obr. 10.9: Závislost absolutní chyby na čase, test č. 1.

V druhém testu se kamera pohybovala po ose x kamery průměrnou rychlostí 2 metry za sekundu. Sklon kamery a výška nad terénem byla přibližně stejná jako u předchozího testu.



Obr. 10.10: Závislost absolutní chyby na čase, test č. 2.

V obou případech se frekvence běhu programu pohybovala v rozmezí 6-7 HZ, systém je tedy schopný v dané konfiguraci zpracovávat data v reálném čase.

10.3 Vlastnosti a omezení systému

Testováním bylo ověřeno, že je systém možné použít pro stabilizaci polohy dronu. Pro úspěšné využití musí však být splněno několik podmínek:

Vhodné prostředí Snímané prostředí musí být dostatečně vizuálně rozmanité pro nalezení příznaků. Při testování například došlo k selhání určování polohy při letu nad stejnorodou asfaltovou plochou. Systém pracuje s předpokladem, že detekované příznaky jsou statické. Pohyb objektů nebo změny prostředí mohou být interpretovány jako pohyb dronu a způsobit selhání systému. Prakticky tyto problémy mohou být způsobeny například snímáním vodních ploch nebo pochybujících se vozidel.

Vhodné vizuální podmínky Pro funkci systému jsou nutná kvalitní obrazová data, je tedy nutná dobrá viditelnost a dostatečné, rovnoměrné osvětlení. Na hranách stínů a odlesků mohou být také detekovány nežádoucí příznaky, které mohou snížit přesnost systému.

Vhodný pohyb dronu S rostoucí výškou nad terénem se snižuje přesnost určení polohy příznaku a tedy přesnost celého systému. Pořízené snímky také nesmí být rozmazané a je nutné aby po sobě pořízených snímcích byly zachycené některé stejné příznaky, tyto podmínky mohou omezovat maximální rychlost a úhlovou rychlost.

Kvalitní informace o orientaci kamery v prostoru. Nepřesné informace o azimutu a náklonu kamery snižují přesnost určování polohy. Vliv těchto odchylek se zvyšuje s rostoucí vzdáleností od detekovaných příznaků, typicky tedy s výškou letu nad terénem.

Vhodné umístění kamerových snímačů Je nutné aby vzájemná poloha a orientace snímačů byla neměnná, dále musí být minimalizovány vibrace kamery. Umístěním kamery na gimbal lze snížit nebo odstranit požadavky na informace o orientaci kamery a omezení maximální úhlové rychlosti. Zvýšení vzdálenosti mezi kamerovými snímači také zvyšuje přesnost při letu ve velké výšce nad terénem.

Závěr

Tato práce se zaměřuje na návrh systému pro stabilizaci polohy dronu pomocí obrazových dat ze dvou kamer. Jsou v ní popsané metody zpracování obrazu a algoritmus pro určení polohy bezpilotního letadla. Dále se práce zabývá výběrem vhodného hardwaru a softwaru pro realizaci systému. Pro detekci a popis příznaků byl vybrán algoritmus ORB. Systém byl s využitím počítače Jetson Nano a stereokamery IMX219-83 realizován a otestován, byly popsány limitace realizovaného systému. Verze implementace určená pro další využití je dostupná na adrese https://github.com/BUT-UARS/drone-visual-pos.

Literatura

[1] GASPAR, Ana Rita, Alexandra NUNES, Andry PINTO a Anibal MATOS. Comparative Study of Visual Odometry and SLAM Techniques. ROBOT 2017: Third Iberian Robotics Conference [online]. Cham: Springer International Publishing, 2018, 2018-12-21, 463-474 [cit. 2023-05-21]. Advances in Intelligent Systems and Computing. ISBN 978-3-319-70835-5. Dostupné z: doi:10.1007/978-3-319-70836-2_38

[2] ARAFAT, Muhammad Yeasir, Muhammad Morshed ALAM a Sangman MOH. Vision-Based Navigation Techniques for Unmanned Aerial Vehicles: Review and Challenges. Drones [online]. 2023, 7(2) [cit. 2023-05-22]. ISSN 2504-446X. Dostupné z: doi:10.3390/drones7020089

[3] HAN, Dianyuan. Comparison of Commonly Used Image Interpolation Methods. Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013) [online]. Paris, France: Atlantis Press, 2013, 2013, - [cit. 2023-05-22]. ISBN 978-90-78677-61-1. Dostupné z: doi:10.2991/iccsee.2013.391

[4] GEDRAITE, Estevao a M. HADAD. Investigation on the effect of a Gaussian Blur in image filtering and segmentation. 2011/01/01, s. -396, 393 s. ISBN 978-1-61284-949-2.

[5] OpenCV: Camera Calibration. OpenCV [online]. [cit. 2023-05-21]. Dostupné z: https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html

[6] MIKSIK, Ondrej a Krystian MIKOLAJCZYK. Evaluation of local detectors and descriptors for fast feature matching. 2012/01/01, s. -2684, 2681 s. ISBN 978-1-4673-2216-4.

[7] LIU, Cuiyin, Jishang XU a Feng WANG. A Review of Keypoints' Detection and Feature Description in Image Registration. Scientific Programming. 2021/12/01, 2021, 1-25. Dostupné z: doi:10.1155/2021/8509164

[8] ROSTEN, E., R. PORTER a T. DRUMMOND. Faster and Better: A Machine Learning Approach to Corner Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence [online]. 2010, 32(1), 105-119 [cit. 2023-01-04]. ISSN 0162-8828. Dostupné z: doi:10.1109/TPAMI.2008.275

[9] HARRIS, C. a M. STEPHENS. A Combined Corner and Edge Detector. Procedings of the Alvey Vision Conference 1988 [online]. Alvey Vision Club, 1988, 1988, 23.1-23.6 [cit. 2023-01-04]. Dostupné z: doi:10.5244/C.2.23

[10] CALONDER, Michael, Vincent LEPETIT, Christoph STRECHA a Pascal FUA. BRIEF: Binary Robust Independent Elementary Features. Computer Vision – ECCV 2010 [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, 2010, 778-792 [cit. 2023-01-04]. Lecture Notes in Computer Science. ISBN 978-3-642-15560-4. Dostupné z: doi:10.1007/978-3-642-15561-1_56

[11] ZIEGLER, Andrew, Eric CHRISTIANSEN, David KRIEGMAN a Belongie SERGE. Locally uniform comparison image descriptor. Advances in Neural Information Processing Systems. 2012, 2012, 1-9. ISSN 1049-5258.

[12] LUCAS, Bruce a Takeo KANADE. An Iterative Image Registration Technique with an Application to Stereo Vision (IJCAI). 81. 1981/04/01.

[13] PEI, Yan, Swarnendu BISWAS, Donald S. FUSSELL a Keshav PINGALI. An elementary introduction to Kalman filtering. Communications of the ACM [online]. 2019, 62(11), 122-133 [cit. 2023-05-22]. ISSN 0001-0782. Dostupné z: doi:10.1145/3363294

[14] OpenCV: Feature Matching. OpenCV [online]. [cit. 2023-01-04]. Dostupné z: https://docs.opencv.org/4.x/dc/dc3/tutorial_py_matcher.html

[15] RUBLEE, Ethan, Vincent RABAUD, Kurt KONOLIGE a Gary BRADSKI. ORB: An efficient alternative to SIFT or SURF. 2011 International Conference on Computer Vision. IEEE, 2011, 2011, 2564-2571. ISBN 978-1-4577-1102-2. Dostupné z: doi:10.1109/ICCV.2011.6126544

[16] MUSSABAYEV, R.R., M.N. KALIMOLDAYEV, Ye.N. AMIRGALIYEV, A.T. TAIROVA a T.R. MUSSABAYEV. Calculation of 3D Coordinates of a Point on the Basis of a Stereoscopic System. Open Engineering [online]. 2018, 8(1), 109-117 [cit. 2023-01-04]. ISSN 2391-5439. Dostupné z: doi:10.1515/eng-2018-0016

[17] OpenCV: Camera Calibration and 3D Reconstruction. OpenCV [online]. [cit. 2023-05-21]. Dostupné z: https://docs.opencv.org/4.x/d9/d0c/group___calib3d.html

[18] Raspberry Pi 4 Model B specifications - Raspberry Pi [online]. [cit. 2023-01-04]. Dostupné z: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

[19] Raspberry Pi 4 Computer Model B. In: Raspberry Pi [online]. 2019 [cit. 2023-05-21]. Dostupné z: https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-4-Product-Brief.pdf

[20] Jetson Nano | NVIDIA Developer [online]. [cit. 2023-01-04]. Dostupné z: https://developer.nvidia.com/embedded/jetson-nano

[21] Jetson Nano | NVIDIA Developer [online]. [cit. 2023-05-21]. Dostupné z: https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[22] IMX 2198-83 Stereo Camera - Waveshare Wiki [online]. [cit. 2023-01-04]. Dostupné z: https://www.waveshare.com/wiki/IMX219-83_Stereo_Camera

[23] OpenCv [online]. [cit. 2023-05-22]. Dostupné z: https://docs.opencv.org/4.x/

[24] Introduction. Mavlink Developer Guide [online]. [cit. 2023-05-21]. Dostupné z: https://mavlink.io/en/

[25] Jetson-utils [online]. [cit. 2023-05-22]. Dostupné z:https://github.com/dusty-nv/jetson-utils

[26] MAVLINK Common Message Set. Mavlink Developer Guide [online]. [cit. 2023-05-21]. Dostupné z: https://mavlink.io/en/messages/common.html

A Obsah elektronické přílohy

Elektronická příloha obsahuje verzi implementace systému určené pro zpracování dat z kamery v reálném čase upravenou pro přehlednost.

```
/ ..... kořenový adresář přiloženého archivu
└── zdrojovy_kod
    ├── main.cpp
    └── config.h
```