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

FÚZE SENZORŮ PRO PLÁNOVÁNÍ POHYBU DRONU

FUSION OF DRONE SENSORS FOR MOTION PLANNING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Jakub Semerád
AUTHOR

VEDOUCÍ PRÁCE — Ing. Jiří Janoušek
SUPERVISOR

BRNO 2022

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: Jakub Semerád ***ID:*** 221014

Ročník: 3 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Fúze senzorů pro plánování pohybu dronu

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s problematikou navigace dronů v neznámém prostředí.
2. Seznamte se s PX4 Autopilot a Robot Operating System(ROS).
3. Navrhněte vhodnou metodu koordinace letu a řešení kolizí.
4. Vyberte potřebné senzorické vybavení pro aplikaci v neznámém prostředí.
5. Realizujte propojení senzorů a řídícího počítače pro řízení dronu.
6. Otestujte a popište výsledky experimentu koordinace dronu v reálném prostředí.

DOPORUČENÁ LITERATURA:

[1] FLOREA, Andrei George a Catalin BUIU. Sensor Fusion for Autonomous Drone Waypoint Navigation Using ROS and Numerical P Systems: A Critical Analysis of Its Advantages and Limitations. 2019 22nd International Conference on Control Systems and Computer Science (CSCS). IEEE, 2019, 2019, , 112-117. ISBN 978-1-7281-2331-8. Dostupné z: doi:10.1109/CSCS.2019.00027

[2] PRASAD, Shashank a Shubhra SINHA. Real-time object detection and tracking in an unknown environment. 2011 World Congress on Information and Communication Technologies. IEEE, 2011, 2011, , 1056-1061. ISBN 978-1-4673-0126-8. Dostupné z: doi:10.1109/WICT.2011.6141394

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 23.5.2022

Vedoucí práce: Ing. Jiří Janoušek

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

Abstrakt

Tato bakalářská práce se zabývá plánováním pohybu dronu v neznámém prostředí. Aby mohl dron vnímat své okolí musí být vybaven potřebným senzorickým vybavením. V práci je diskutováno možné senzorické vybavení vhodné pro koordinaci letu a řešení kolizí. Také jsou zde popsány a využity softwary pro zpracování dat ze senzorů a plánování pohybu dronu. Pro experiment v této práci byl využit LiDAR a stereokamera. Data z obou senzorů jsou zde prostřednictvím softwaru ROS spojena do jednoho celku a zobrazována v simulačním prostředí RViz. Tato data jsou vhodná pro následné využití při plánování pohybu dronu a řešení kolizí.

Klíčová slova

Dron, ROS, LiDAR, Stereokamera, PX4, UAV

Abstract

This bachelor thesis deals with drone flight planning in an unknown environment. In order for a drone to be able to perceive its environment, it must be equipped with the necessary sensory equipment. This thesis discusses possible sensor equipment suitable for flight coordination and collision avoidance. Also, software for sensor data processing and drone motion planning are described and used here. LiDAR and depth camera were used for the experiment in this thesis. In the experiment the data from both sensors are fused together through ROS software and displayed in RViz simulation environment. This data are suitable for subsequent use in drone motion planning and collision avoidance.

Keywords

Drone, ROS, LiDAR, Depth camera, PX4, UAV

Bibliografická citace

SEMERÁD, Jakub. Fúze senzorů pro plánování pohybu dronu [online]. Brno, 2022 [cit. 2022-03-27]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/142032. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Jiří Janoušek.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	*Jakub Semerád*
VUT ID studenta:	*221014*
Typ práce:	*Bakalářská práce*
Akademický rok:	*2021/22*
Téma závěrečné práce:	*Fúze senzorů pro plánování pohybu dronu*

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 jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

.

V Brně dne: 17.5.2022

podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Jiřímu Janouškovi za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne: 17.5.2022

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

Za posledních několik let se použití bezpilotních letounů (UAV) stalo obvyklým ve velké spoustě aplikací. V současné době jsou UAV nejvíce využívány pro výškové inspekce v infrastruktuře, pořizování snímků či videí nebo pro zlepšení zemědělství. Probíhající technický vývoj ve velikosti, době letu a spolehlivosti umožní využití bezpilotních letounů v ještě širší oblasti. Již dnes existuje několik prototypů pro doručování zboží nebo dokonce pro přepravu osob. I když by už bylo možné nasadit takovéto prototypy do reálného provozu, není tomu tak, a to z jednoduchého důvodu. Vzdušný prostor je komplexní a velmi rychle se dynamicky mění. Zároveň ho nevyužívají jenom bezpilotní letouny, proto kdyby došlo k jakékoliv havárii následky by mohly být fatální. Aby se předešlo nebezpečnému pohybu těchto UAV, podléhá jejich provoz přísným legislativním pravidlům, a právě z tohoto důvodu nasazení jakékoliv aplikace využívající autonomní letouny je velmi složité.

Pro vyšší použitelnost a bezpečnost jsou bezpilotní letouny vybaveny senzory pro monitorování prostředí. Obvykle drony nesou měřič vzdálenosti či kameru, nebo pro vyšší přesnost využívají oba tyto senzory. Jedním z důležitých aspektů tohoto monitorování je detekce překážek v dráze letu tak, aby se předešlo kolizím a mohla být nastavena nová dráha letu dronu. Tento postup vede ke zvýšení autonomie UAV, avšak přináší náročnost na palubní výpočty, plánování trajektorie a komunikaci s pozemní stanicí.

Zejména v posledních letech exponenciálně roste zájem výzkumníků i amatérských uživatelů o další rozšíření schopností bezpilotních letadel, což povede k jejich větší spolehlivosti a díky tomu budou zcela jistě nasazovány do více a více aplikací.

Tato práce se zabývá zvýšením stupně autonomie bezpilotního letadla, konkrétně multikoptéry pomocí fúze dat dvou na sobě nezávislých senzorů. Cílem je zpracovávat data z lidaru a stereokamery a pomocí nich vytvořit situační přehled v bezprostředním okolí dronu. Očekávaným výstupem je dosažení řešení, které bude schopné předejít možným kolizím, které mohou nastat s překážkami v trase letu.

1. NAVIGACE DRONŮ V NEZNÁMÉM PROSTŘEDÍ

Dron nebo obecně robotický systém lze prohlásit za plně autonomní, pokud se řídí a rozhoduje sám bez zásahu člověka. Může být i částečně autonomní, a to právě v závislosti na míře potřeby zásahu jeho pilota. Přesné rozdělení je popsáno v této kapitole a zobrazeno na obrázku 1. Všechny letouny, a to jak ty autonomní nebo ty řízené čistě pilotem podléhají legislativním podmínkám, které jsou taktéž v této kapitole uvedeny.

Na závěr se tato kapitola zabývá principem navigace dronů v neznámém prostředí a samotným využitím těchto autonomních prostředků, a to jak využitím současným, tak možným využitím v budoucnosti.

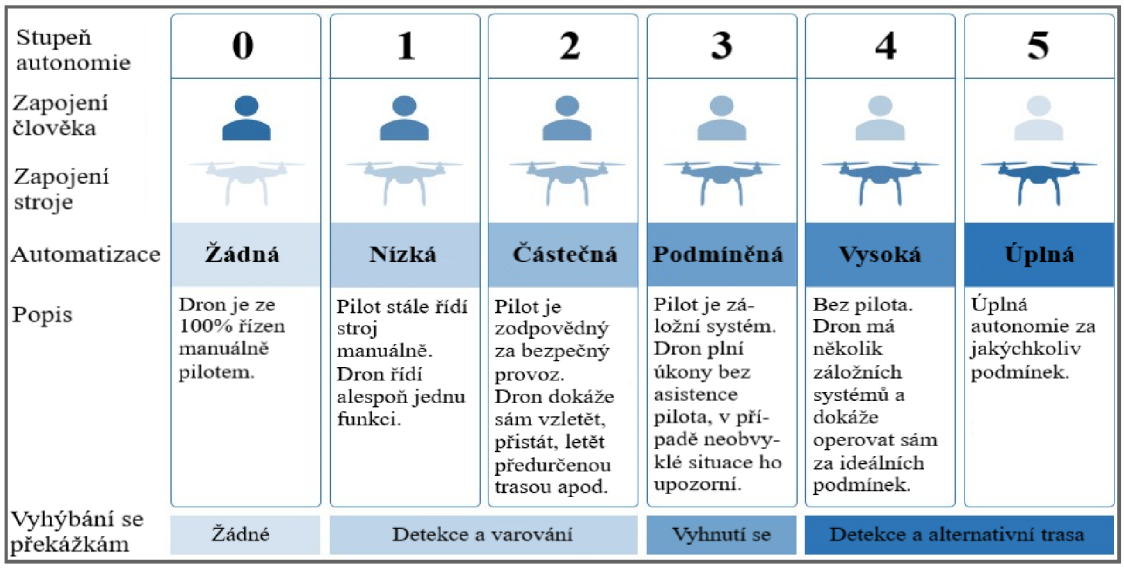
1.1 Autonomní systémy

Autonomní systém, v tomto případě dron, je schopen se sám rozhodovat bez zásahu člověka. Dokáže reagovat na nečekanou a nestandardní situaci. Využívá k tomu potřebné algoritmy a prvky umělé inteligence. Nezbytnou součástí autonomního dronu je senzorické vybavení, bez kterého by nedokázal identifikovat měnící se podmínky v jeho okolí. Autonomní drony nejsou momentálně povoleny v „otevřené“ kategorii provozu. Jednotlivé kategorie provozu autonomních dronů jsou podrobněji popsány níže v této kapitole.

Oproti tomu automatické drony mohou létat a plnit potřebné úkoly v některých případech částečně sami, avšak když nastane nestandardní situace je zapotřebí zásah pilota.

1.1.1 Stupeň autonomie dronu

Stupeň autonomie dronu lze rozdělit do šesti úrovní podle míry automatizace (viz. obrázek 1). V nulté úrovni je dron z plné části řízen manuálně pilotem. U první úrovně pilot pořád řídí let manuálně, avšak dron dokáže zastat jednu funkcionalitu, kterou může být například udržování výšky, pomoc s navigací či držení pevně dané pozice v prostoru. Na druhé úrovni je většina současných dronů. Pilot už nemusí manuálně řídit každý pohyb, ale musí být připravený kdykoliv převzít kontrolu. Drony na této úrovni dokážou sami vzlétnout a přistát, letět přednastavenou trasou či mapovat nebo fotografovat prostředí. Ve třetí úrovni letoun zvládne letět bez asistence pilota, ten však musí být připraven zasáhnout. Ale oproti předchozímu stupni ho dron dokáže upozornit v případě, kdy by měl pilot zareagovat. Čtvrtý stupeň se vyznačuje tím, že dron dokáže létat sám za ideálních podmínek. Při detekování překážky je schopen si nastavit alternativní trasu a překážce se co nejefektivněji vyhnout. Pochopitelně pátá poslední úroveň neboli plná autonomie popisuje letoun, který se kompletně pilotuje sám za jakýchkoliv podmínek. V momentální chvíli takové drony ještě neexistují [1].



Obrázek 1 Stupně autonomie dronu [1]

1.2 Provoz bezpilotních letounů

Provoz bezpilotních letadel je z pohledu legislativy rozdělen do třech hlavních kategorií.

„Otevřená“ (Open) kategorie je kategorií provozu bezpilotních systémů, u kterých s ohledem na související rizika není vyžadováno předchozí povolení příslušného úřadu, ani prohlášení provozovatele UAS před uskutečněním provozu. Pod otevřenou kategorii spadají i tři podkategorie, které jsou podrobněji popsány na obrázku 2.

Podkategorie „otevřené“ kategorie provozu	Štítek s označením třídy typu dronu
A1 **Urbanistické oblasti, ale ne nad davy, nebo mimo urbanistické oblasti**	Štítek s označením třídy C0, C1
	Soukromě zhotovený dron s MTOM < 250 g a rychlostí < 19 m/s
	Dron bez štítku s označením třídy s MTOM < 500 g (do 1. ledna 2023)
	Dron bez štítku s označením třídy s MTOM < 250 g včetně paliva a užitečného zatížení. (od 1. ledna 2023)
A2 **Urbanistické oblasti při udržování nejméně 30 m (ve zvláštních případech až 5 m) od lidí, nebo mimo urbanistické oblasti**	Štítek s označením třídy C2
	Dron bez štítku s označením třídy s MTOM < 2 kg (do 1. ledna 2023) (minimální vzdálenost od osob je v tomto případě navýšena na 50 m)
A3 **Mimo urbanistické oblasti**	Štítek s označením třídy C2, C3, C4
	Soukromě zhotovený dron s MTOM < 25 kg rychlost < 19 m/s
	Dron bez štítku s označením třídy s MTOM < 25 kg

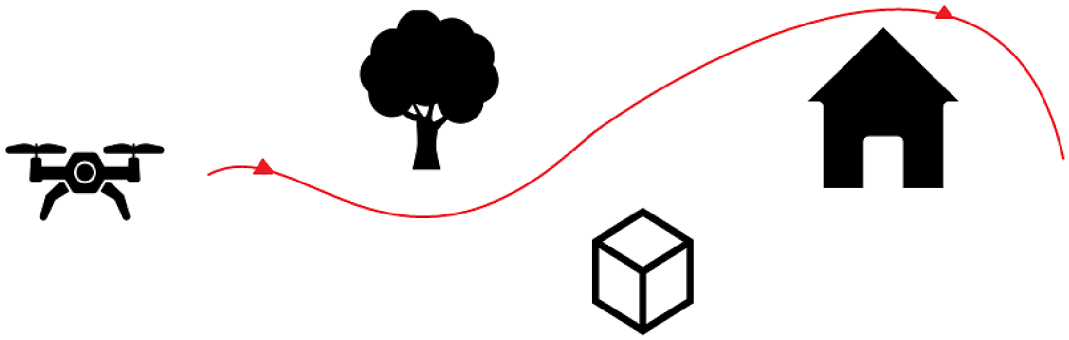
Obrázek 2 Podkategorie otevřené kategorie provozu bezpilotních letounů [2]

„Specifická“ (Specific) kategorie je kategorie provozu bezpilotních systémů, u kterých je s ohledem na související rizika vyžadováno povolení příslušného úřadu (v případě České republiky oprávnění k provozu vydané Úřadem pro civilní letectví) před uskutečněním provozu, s uvážením zmírňujících opatření identifikovaných v posouzení provozního rizika.

„Certifikovaná“ (Certified) kategorie je kategorie provozu bezpilotních systémů, u kterých je s ohledem na související rizika vyžadována certifikace bezpilotního systému, osvědčení způsobilosti dálkově řídícího pilota a schválení provozovatele příslušným úřadem, aby byla zajištěna odpovídající úroveň bezpečnosti [2].

1.3 Autonomní navigace

Autonomní navigace zjednodušeně znamená úprava letové dráhy v závislosti na měnícím se prostředí (naznačeno na obrázku 3) a lze ji dosáhnout pomocí algoritmů, které dronu umožňují vnímat své okolí a na základě toho se rozhodovat. Vědecká literatura na toto téma se velmi rychle rozšiřuje. Existuje velké množství algoritmů, které jsou stále odolnější vůči různým překážkám a dalším složitým změnám v prostředí. Obecně neexistuje žádné správné řešení pro navigaci autonomních prostředků. Je to velmi komplexní a náročný úkol, kde je potřebné navrhnout řešení v závislosti na daném použití. Není možné například využít informace pouze z GPS, ty mohou být sice jednoduše k dispozici, ale neposkytují dostatečně přesná data pro autonomní řízení dronu. Algoritmy použité pro efektivnější autonomní řízení se hlavně spoléhají na data z více senzorů umístěných přímo na autonomním prostředku.

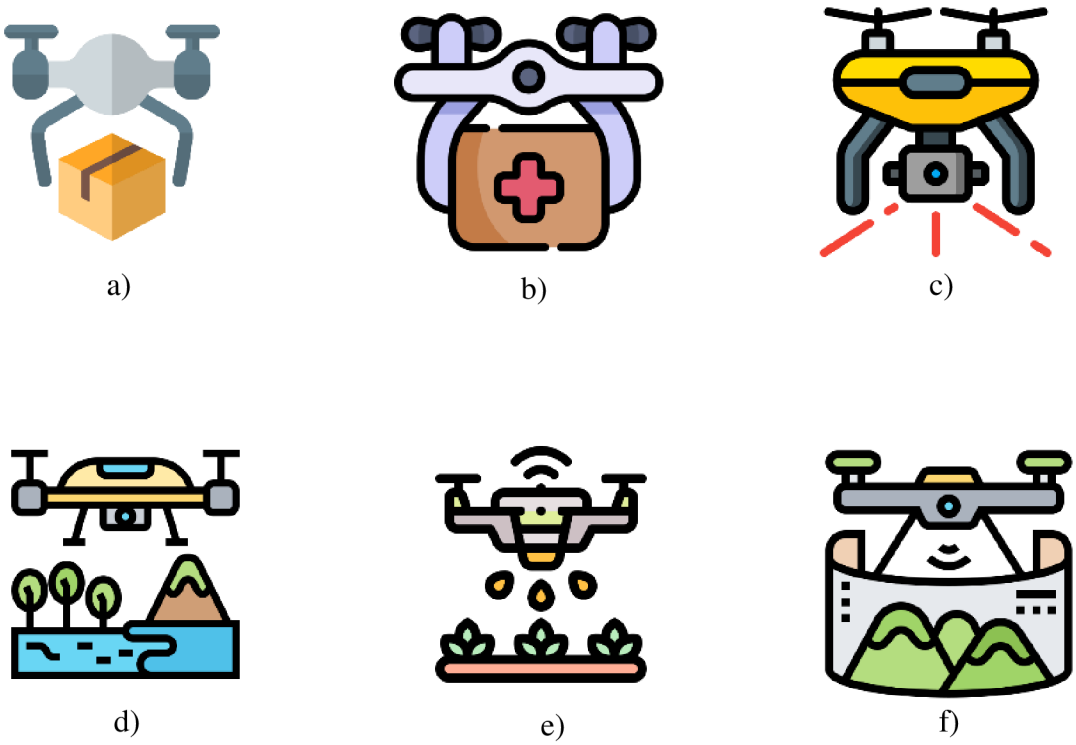


Obrázek 3 Autonomní navigace dronu

1.4 Využití autonomních dronů

Autonomní drony mají již v dnešní době zastoupení v širokém spektru aplikací. Spadá mezi ně inspekce infrastruktury (obrázek 4 - část c), terénní průzkumy (obrázek 4 - část d), sběr vědeckých dat a monitorování pro zabezpečení prostorů. Kromě toho jsou vhodné pro transport a dodávky zboží (obrázek 4 - část a), povrchovou těžbu, mapování (obrázek 4 - část f), precizní zemědělství, kde se správnou technologií mohou ušetřit čas a peníze, zvýšit efektivitu a snížit množství pracovní síly. Autonomní bezpilotní letouny mohou například postřikovat pole, monitorovat stav přírodních biotopů, dodávat léky během přírodních katastrof (obrázek 4 - část b), nebo se mohou dostat do oblastí, kam je jinak obtížné se s dronem, který je řízený pilotem

dostat, což zkracuje dobu letu a minimalizuje chyby ve srovnání s létáním, které řídí pilot. V současné chvíli využívá částečně autonomní drony firma Flytrex, která dodává spotřební zboží a potraviny. NVIDIA postavila dron, který je schopný částečně létat bez GPS. Dokáže tedy létat například přes lesní cesty, kde by mohl hledat ztracené turisty, spadlé stromy či mapovat turistické stezky. Firma SenseFly vyvinula mini-drony pro profesionální mapování a fotografování. Nelze opomenout ani známého výrobce dronů DJI, který se svým nejnovějším průmyslovým dronem Matrice 30 dokáže mapovat prostředí, zaznamenávat scénu klasickou barevnou kamerou či termokamerou. Zároveň je pak vybaven i několika senzory, které slouží hlavně pro detekci překážek v dráze letu, což z něho dělá dron, u kterého téměř nemůže nastat situace, že by havaroval.



Obrázek 4 Využití autonomních dronů

2. SYSTÉMY PRO PLÁNOVÁNÍ POHYBŮ DRONŮ

Tato kapitola se podrobně věnuje všem používaným softwarům v oblasti plánování pohybu dronu. Hlavní částí v této kapitole je Robot Operating System (ROS), kterého je v dnešní době využíváno téměř v každé aplikaci v oblasti plánování pohybu autonomních prostředků. Další nezbytnou částí pro řízení pohybu dronu je autopilot. Pro software vytvářející funkci autopilotu jsou primárně používané dva systémy, a to PX4 Autopilot a ArduPilot.

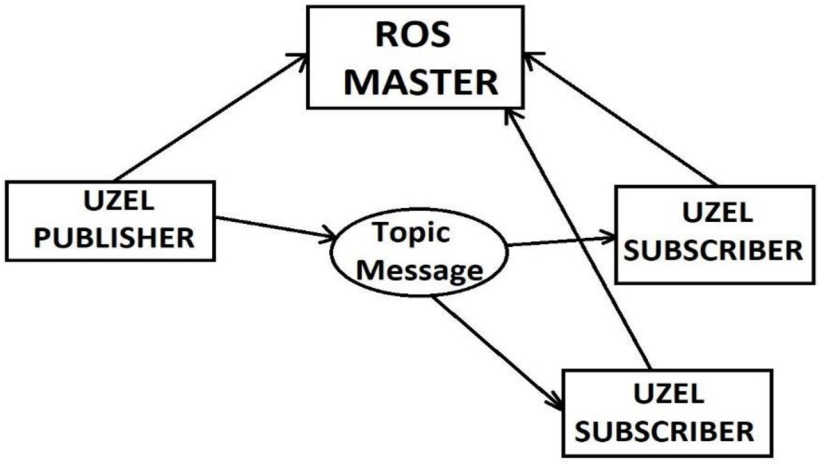
Před nasazením autonomní navigace na reálný hardware je vhodné nejprve provést potřebné simulace, což vede později k lepším dosaženým výsledkům při reálných experimentech a zároveň se tak předejte jakémukoliv nechtěnému chování robotického prostředku a možné následné havárie. Softwary pro takovéto simulace jsou taktéž popsány v této kapitole.

2.1 Robot Operating System (ROS)

Nejedná se o skutečný operační systém, ale o open source framework, který je převážně určen pro Unixové platformy. Primárně je tedy testován na Ubuntu či Mac OS X.
V dnešní době je několik dostupných distribucí ROSu. V této práci bylo využito distribuce ROS Melodic, která běží na Ubuntu 18.04. Dále je často využíván ROS Noetic, který je určený pro Ubuntu 20.04. ROS je momentálně rozdělen na ROS 1 a ROS 2. Hlavním rozdílem mezi těmito verzemi je, že ROS 2 dokáže pracovat v reálném čase a také je možné s ním pracovat v operačním systému Microsoft Windows. ROS 1 a ROS 2 nejsou mezi sebou kompatibilní, proto se aktuálně udržují obě platformy. Pod ROS 1 spadají právě Melodic a Noetic. Nejnovější verzi ROSu 1 verzi Noetic skončí podpora v květnu roku 2025 a bude tak nutné přejít kompletně na ROS 2. ROS 2 je možné využít například v distribuci Foxy Fitzroy.

Základem ROSu jsou uzly (anglicky Nodes), které plní funkci jednotlivých procesů. Tyto Nody je možné psát ve spoustě moderních programovacích jazyků jako Python, C++, Lisp a experimentálně v Javě. Uzly mohou být seskupeny do tzv. balíčků (anlicky Packages), což umožňuje je lehce distribuovat. Každý balíček obsahuje XML soubor, který v sobě definuje závislosti balíčku na jiných balíčcích. Dále je ROS tvořen zprávami (anglicky messages) a službami (anglicky services). Topiky jsou jakési mosty, přes které si dané programy (Nody) vyměňují svoje zprávy. Typickým topikem může být například topik scan, jenž představuje 2D mračno bodů získané z laserového skeneru vzdálenosti. Tyto topiky je možné si po pozdější zpracování ukládat do tzv. ROS bagu.

Robot obvykle provádí procesy jako zpracování dat ze senzorů, manipulace, vyhýbání se překážkám, plánování trasy a další. Pro vhodné volání procesů, které jsou popsány v uzlech, je použit ROS Master nebo také jinak nazýván ROS Launch (viz. obrázek 5). Díky tomu je možné docílit plynulého chodu procesů, které robot vykonává [3][19].



Obrázek 5 Příklad propojení uzlů s ROS Masterem

ROS je tedy hlavně využíván k propojení dílčích programů do komplexního celku. Bez použití ROSu by bylo zapotřebí vymyslet, jak budou všechny dílčí programy komunikovat. Například by byla možnost sdílet paměť, pipovat, posílat si IP zprávy nebo používat DBus. Všechny tyto techniky jsou možné, ale oproti jednoduchému využití softwaru ROS jsou „programátorsky“ dost náročné [20]. Právě proto byl Robotic Operating System využit pro zpracování dat v tomto experimentu.

2.2 Robotické simulátory

Robotické simulátory umožňují design, simulace a testování robotických aplikací v příslušném fyzickém prostředí a to nezávisle na dostupnosti reálného hardwaru. Díky tomu robotické simulátory mohou šetřit čas vývoje a jeho náklady.

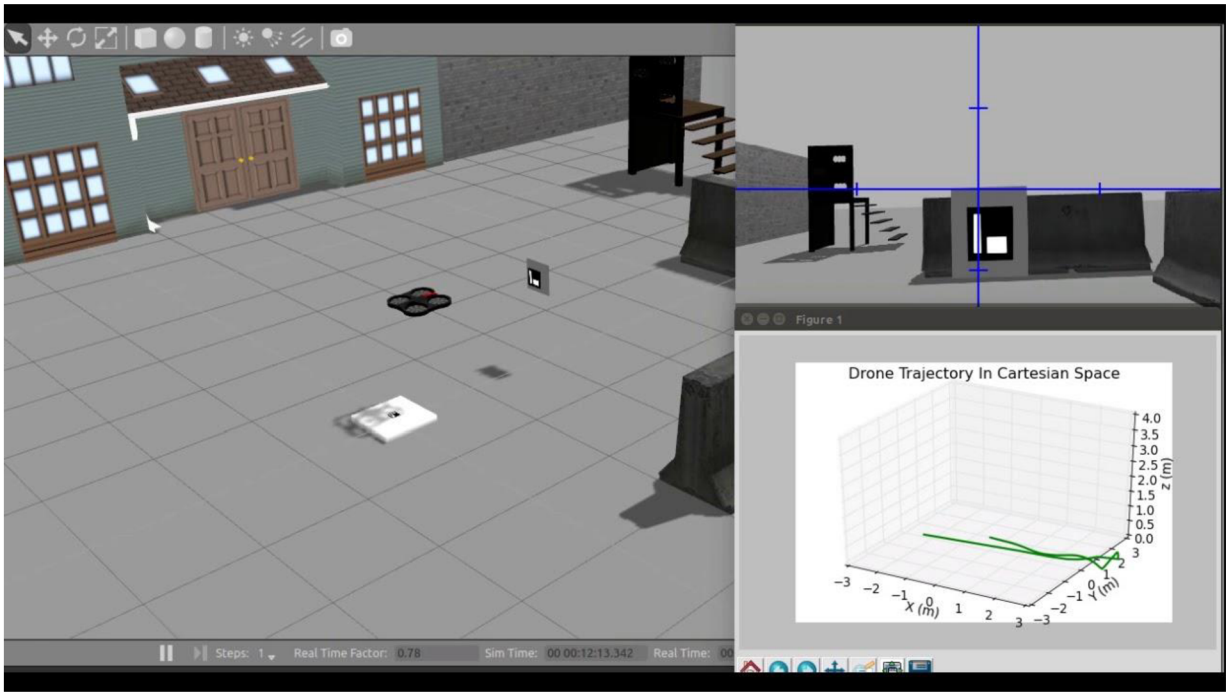
Druhů robotických simulátorů je několik. Je možné je rozdělit podle několika vlastností. Například pro jaký operační systém jsou vhodné, jaký je jejich hlavní programovací jazyk, kolik dimenzí dokáží simulovat (3D a 2D) nebo jak dobře jsou kompatibilní se softwarem ROS. Mezi ty nejznámější patří například Gazebo, RViz, MORSE, V-REP či Unity. Pro účely této práce bylo využito robotických simulátorů Gazebo a RViz.

2.2.1 Gazebo

Gazebo je software, který umožňuje 3D simulace robotů se senzory, a to jak ve vnitřím, tak ve venkovním prostředí. Příklad simulace dronu v prostředí Gazebo je zobrazen na obrázku 6. Tento software má architekturu typu Klient/Server a disponuje komunikačním modelem Publisher/Subscriber. Uživatelé Gazeba mají přístup k jeho datům díky sdílené paměti. Každý simulovaný objekt může být spojen s jedním nebo více řídícími systémy pro řízení a generování jeho stavu. Data, která řídící systémy generují jsou nahrávány do sdílené paměti pomocí rozhraní. Tato rozhraní dokážou data ze sdílené paměti i číst, což umožňuje komunikaci mezi softwarem ROS a simulátorem Gazebo nezávisle na programovacím jazyce a použitém operačním systému. Pro simulovaní dynamických procesů Gazebo využívá vysokovýkonové

fyzikální frameworky jako ODE, Bullet, Simbody a DART. Pro vykreslování 3D grafiky Gazebo využívá framework OGRE (Object-Oriented Graphics Rendering Engine).

Použití simulátoru Gazebo v softwaru ROS je velmi výhodné, protože se využívá pouze jedno rozhraní, tedy je možné řídit simulaci a vyvíjet robotické systémy v jednom systému [4] [5].

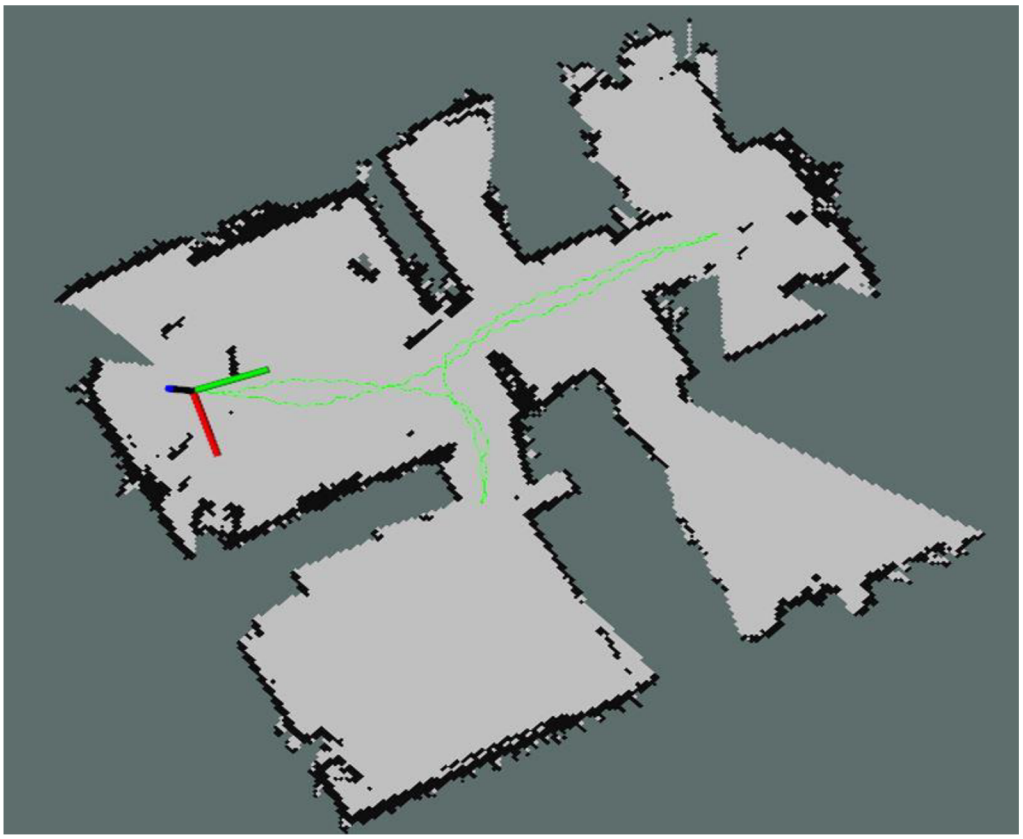


Obrázek 6 Simulace dronu v prostředí Gazebo [21]

2.2.2 RViz

Rviz je vizualizační nástroj, který je dodáván jako součást ROSu. Jedná se o aplikaci, která dokáže poslouchat širokou paletu předdefinovaných ROS zpráv a vizualizovat je v 3D grafickém prostředí. Obvykle lze RViz použít pro vizualizaci pointcloudů (mračen bodů například z LiDARu), obrázků z kamery, vykreslování geometrických primitiv v prostoru, vizualizace map, vykreslení trajektorie apod [20].

Příklad vykreslení mapy z dat získaných pomocí LiDARu v prostředí RViz lze vidět na obrázku 7.



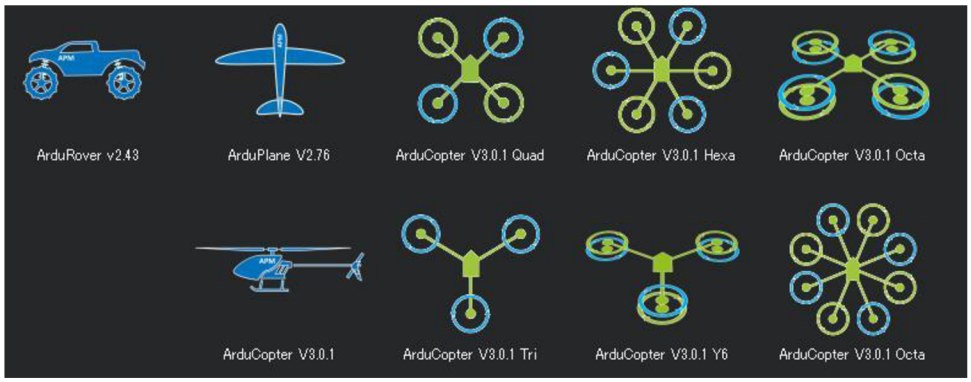
Obrázek 7 Mapa prostředí vizualizovaná v simulačním nástroji RViz

2.3 Autopiloty

Mezi nejvíce rozšířené autopiloty zcela jistě patří ArduPilot a PX4 Autopilot. Oba softwary jsou typu open-source a jejich použití je velmi vhodné pro řízení autonomních prostředků bez nutnosti lidské obsluhy. Jsou plně kompatibilní se simulačním prostředím Gazebo. Pro komunikaci s pozemní stanicí využívají protokol MAVLink. Oba zmíněné autopiloty disponují velmi rozsáhlou dokumentací, která je volně dostupná a pořád se rozšiřuje. Odlišují se v typu licence. U ArduPilot je licence GPL, což znamená, že osoby, které upravují a následně prodávají aplikaci využívající ArduPilot jsou povinné zpřístupnit své úpravy. U PX4 je licence BSD, což znamená přesný opak, tedy nemusejí být zpřístupněny úpravy.

2.3.1 ArduPilot

ArduPilot je možné využít pro ovládání téměř jakéhokoliv leteckého či pozemního dopravního prostředku (viz. obrázek 8). Konkrétně jím může být letadlo, kluzák, multikoptéra, vrtulník, plachtenice, motorový člun, ponorka nebo pozemní vozidlo.



Obrázek 8 Možnosti využití ArduPilot [22]

Pozemní stanice v případě ArduPilot běží na softwaru Missin Planner nebo APM Planner a opět při komunikaci využívá protokol MAVLink.

Struktura softwaru ArduPilot je rozdělena do 5 částí: adresář vozidla, knihovny, vrstva hardwaru, adresáře nástrojů a podpora externího kódu. Adresář vozidla je nejvyšší adresář, který definuje firmware pro jednotlivé typy vozidel a spolu se soubory .cpp taktéž obsahuje wscript, v němž jsou definované závislosti knihoven. Knihovny jsou společné pro všechny typy vozidel a obsahují například ovladače snímačů nebo kód pro regulátory. Vrstva hardwaru v sobě zahrnuje způsob, jak zajistit přenositelnost softwaru ArduPilot na různé platformy. V adresáři libraries/AP_HAL je nejvyšší úroveň AP_HAL, která definuje rozhraní, jež má zbytek kódu ke specifickým funkcím desky, a pak je zde podadresář AP_HAL_XXX pro každý typ desky, například AP_HAL_AVR pro desky založené na AVR, AP_HAL_PX4 pro desky Pixhawk a AP_HAL_Linux pro desky založené na Linuxu. Adresáře nástrojů jsou podpůrné adresáře. Například adresář tools/autotest poskytuje infrastrukturu pro autotesty na webu autotest.ardupilot.org a adresář tools/Replay poskytuje nástroj pro přehrávání protokolů [22].

2.3.2 PX4 Autopilot

Projekt PX4 byl zahájen již v roce 2009. Tento open-source software se převážně orientuje na nízkonákladové autonomní letadla a je neustále vyvíjen aktivní celosvětovou komunitou. Je dodáván jako součást letového počítače Pixhawk, kde plní funkci firmwaru. Díky rychlé přizpůsobivosti na tento software a snadnému použití je PX4 velmi oblíbený ve výzkumné oblasti [29].

Celý firmware PX4 je založen na principu Publisher/Subscriber. Publish/Subsribe je princip odesílání zpráv, kde odesílatel zpráv (publisher), neposílá zprávu rovnou příjemci (subsriber), ale centralizovanému zprostředkovateli, který následně zprávy distribuuje příjemcům. Použití tohoto vzoru znamená, že:

- je asynchronní a data se aktualizují hned, když jsou k dispozici
- všechny operace a komunikace běží paralelně
- komponenty systému mohou přijímat data odkudkoliv

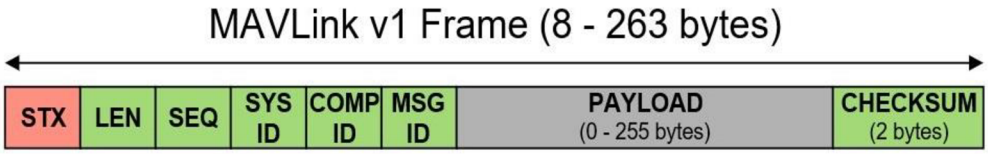
Entity, které vytvářejí, posílají a přijímají zprávy v PX4 se nazývají moduly. Firmware PX4 si lze představit jako systém složený z různých modulů, takže každý z nich je součástí systému

a je určený k výpočtu specifické úlohy. Moduly spolu komunikují prostřednictvím sběrnice zpráv publish/subscribe s názvem uORB, což je asynchronní API pro publikování a přihlášení zpráv používaných pro komunikaci mezi vlákny a procesy.

Firmware PX4 se skládá ze dvou hlavních vrstev: middleware a flight stack. Middleware obsahuje především ovladače pro vestavěné senzory, komunikaci s pozemní stanicí a sběrnici zpráv uORB. Naproti tomu flight stack je soubor naváděcích, navigačních a řídicích algoritmů pro autonomní drony. U systému PX4 obvykle na pozemní stanici běží software QGroundControl [30].

2.4 MAVLink

MAVLink je knihovna zpráv pro komunikaci mezi dronem a pozemní stanicí. Může přenášet údaje o telemetrii, stavu mise, stavu senzorů a aktorů. Skládá se ze specifikací sady zpráv, které jsou definované v souborech XML. Tyto specifikace je možné upravit. Například ArduPilot používá celou sadu zpráv a přidává k ní některé další, které jsou zaměřeny především na ovládání jeho přidaných senzorů a funkcí. Python nástroje je převádějí na vhodný zdrojový kód pro podporované jazyky. Existuje velké množství skriptů v programovacím jazyce Python, které slouží jako příklady a nástroje pro práci s daty MAVLink protokolu. MAVlink je vhodný pro použití v aplikacích s omezenou komunikační šířkou pásma. Jeho referenční implementace v jazyce C je vysoce optimalizovaná pro systémy s omezenou RAM a flash pamětí. Je dlouhodobě osvědčený v praxi a je využíván v mnoha aplikacích, kde slouží jako rozhraní mezi komponenty od různých výrobců [6].



Obrázek 9 Rozložení bytů zprávy v MAVLinku [6]

Na obrázku 9 jsou zobrazeny jednotlivé byty zprávy MAVlink protokolu. Minimální délka takové zprávy je 8 bytů a maximální je 255 bytů. Jednotlivé byty jsou popsány v tabulce 1.

Tabulka 1 Popis jednotlivých bytů zprávy MAVLinku [6]

Pozice bytu	Název	Obsah	Popis
0	STX	Začátek zprávy	Označuje začátek nové zprávy.
1	LEN	Délka zprávy	Udává délku zprávy.
2	SEQ	Pořadí zprávy	Definuje pořadí zprávy.
3	SYS	ID systému	Udává ID odesílajícího sytému.
4	COM	ID komponenty	Udává ID odesílající komponenty.
5	MSG	ID zprávy	Označuje ID zprávy.
6 až n+6	PAYLOAD	Data	Jednotlivá data zprávy.
n+7 až n+8	CKA+CKB	Kontrola	Kontrolní součet a případná oprava chyb.

2.5 MAVROS

MAVROS je balíček v ROSu, který společně s MAVLink protokolem umožňuje nastavování parametrů a odesílání příkazů přímo z PC. MAVROS a MAVLink tedy tvoří jakýsi most mezi PC, pozemní stanicí a autopilotem. Uživateli je tedy umožněno programovat a přizpůsobovat parametry autopilota přímo v prostředí ROS a nahradit tak všechny funkce softwaru pozemní stanice.

MAVROS má několik základních příkazů, které se využívají pro komunikaci s autopilotem. Takovými příkazy jsou například /mavlink/from a /mavlink to, které umožňují odesílání dat z a do autopilota. Dále to může být příkaz /mavros/state, který slouží ke kontrole stavů a parametrů autopilota. Důležitými příkazy jsou mavsafety a mavsys. Příkaz mavsys je využíván v případě, kdy je zapotřebí řídit autopilot přímo z konzole. Mavsafety umožňuje například odjištění vozidla, aby se mohlo pohybovat. U příkazů mavsafety a mavsys je nezbytné dodržovat jejich pořadí. Například když je nutné změnit parametry režimu pomocí příkazu mavsys, nejdříve se musí provést odjištění robota pomocí příkazu mavsafety [7].

2.6 Dronekit

Jedná se o open source projekt, který je vyvíjen širokou komunitou. Díky tomu jsou jeho zdrojové kódy jednoduše dostupné. Může být implementován na všechny běžné operační systémy a je kompatibilní s robotickými prostředky, které používají MAVLink protokol. DroneKit umožňuje vývojářům vytvářet aplikace, které komunikují s autopilotem. Použití DroneKitu může významně zlepšit autonomní režim robotického systému, a to díky přidání algoritmů umělé inteligence, počítačového vidění, plánování pohybu nebo 3D modelování. DroneKit může být současně použit pro aplikace použité v pozemních stanicích. API komunikuje s robotickými prostředky prostřednictvím MAVLink protokolu, což umožňuje přístup k telemetrii, stavu či parametrům robota a dovoluje plánování misí nebo přímé řízení jeho pohybu [8].

2.6.1 DroneKit SITL

SITL (z anglického výrazu software in the loop) je velmi výhodným nástrojem pro testování a simulace bez použití reálného dronu a hardwaru. Přestože se jedná o relativně experimentální nástroj, tak SITL dokáže pracovat na všech běžných operačních systémech. SITL může být nainstalován na stejném PC jako DroneKit nebo na jakémkoliv PC ve stejné síti. Tento software dokáže velmi dobře provádět testovací kód, kterým je popsáno chování dronu a následně je možné sledovat, co se děje se simulovaným dronem přímo na obrazovce pozemní stanice [29]. Instalace SITLu se provádí pomocí nástroje Python pip, což je instalační program pro balíčky pythonu. Po instalaci se musí uživatel rozhodnout jaký druh letounu bude chtít simulovat, může to být například koptéra, letadlo a další [23].

3. SENZORICKÉ VYBAVENÍ PRO BEZKOLIZNÍ POHYB DRONŮ

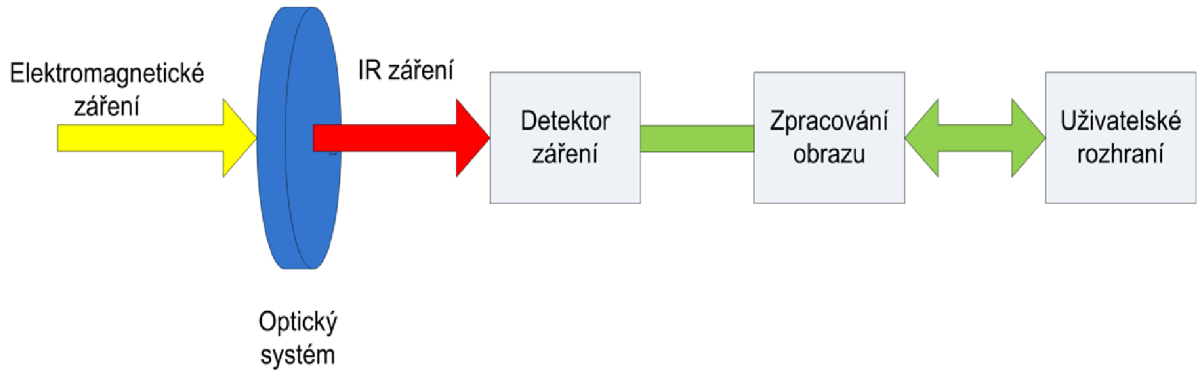
Aby se jakýkoliv autonomní robotický prostředek mohl pohybovat sám v dynamicky se měnícím prostředí, musí snímat svoje okolí, vyhodnocovat nebezpečí srážky s možnou překážkou a plánovat svoji trasu pohybu. Takového snímání se dá docílit několika senzorickými systémy. První velkou kategorií jsou kamery. Do této kategorie však nepatří jen obyčejné digitální barevné kamery, ale i infrakamery, které dokážou vyhodnocovat teplotu povrchu objektů, multispektrální kamery a také stereokamery, díky kterým se dá získat hloubka obrazu a tím i vzdálenost objektů na snímku. Dalším velmi často používaným senzorem je laserový měřič, kterým samozřejmě nelze získat obraz okolí, ale pouze vzdálenosti překážek od robotického vozidla. Pokud se takový laserový měřič pomocí motoru otáčí o celých 360 stupňů, jedná se o LiDAR, který velmi rychle a přesně dokáže zmapovat vzdálenosti objektů ve všech směrech od vozidla či letounu.

LiDAR i kamerový systém jsou vhodnými senzory pro bezkolizní pohyb dronů, ale mají i svoje nevýhody. Podrobně jsou popsány v této kapitole. Protože fungují na různých principech, tak jeden senzor může například lépe snímat okolí ve tmě a druhý zase při detekci objektů s téměř dokonale odrazivým povrchem. Proto není ideální pro komplexní řízení pohybu dronu v neznámém prostředí využít pouze jeden z těchto senzorů, ale použít je oba a data z nich získaná sloučit do jednoho celku a vytvořit tak senzorickou fúzi.

3.1 Kamery

3.1.1 Infrakamera

Detekce objektů pomocí infrakamery vychází z jednoduchého principu. Každý objekt, který je teplejší jak absolutní nula, tedy jak -273,15 °C vyzařuje elektromagnetické záření. Ve skutečnosti je to každý předmět ve známém vesmíru. Pomocí infrakamery se následně měří intenzita dopadající záření. Samotná infrakamera se skládá z několika částí (viz. obrázek 10).



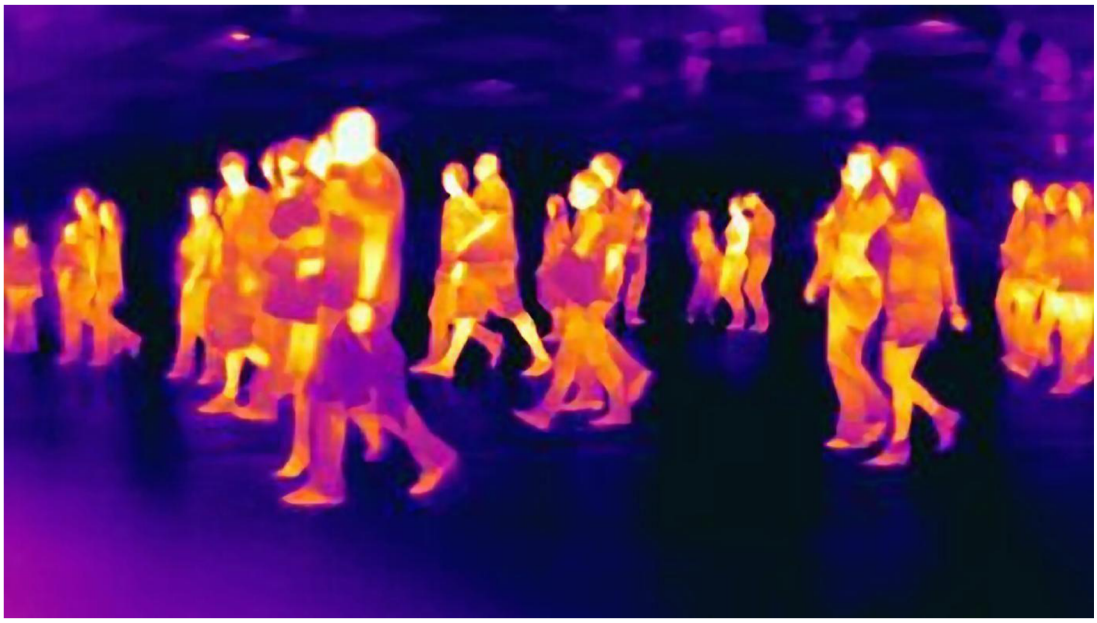
Obrázek 10 Blokové schéma infrakamery [13]

První částí je optický systém, na který přímo dopadá elektromagnetické záření, které emitují měřené objekty. Tento optický systém je navržený tak, aby propustil z celého spektra elektromagnetického záření pouze záření požadovaných vlnových délek, tedy infračervené záření. Samotné infračervené záření následně dopadá na detektor, který obvykle mění elektrický odpor v závislosti na intenzitě dopadajícího záření. Detektory mohou být chlazené či nechlazené. Chlazené detektory umožňují přesnější měření, avšak jsou značně náročnější na provedení, což se projeví na celkové ceně infrakamery. Hodnoty elektrického odporu jsou následně digitalizovány a převedeny na tzv. termogram, který je složený z pixelů, kde jednotlivé pixely přímo udávají hodnoty povrchové teploty snímaného objektu.

Velkou výhodou je možnost sledování tepelných dějů přímo v reálném čase. Je ale nutné si uvědomit, že infrakamera detekuje pouze tepelné děje na povrchu objektu.

Aplikací, kde je využívána infrakamera, je v dnešní době nespočet. Mezi takové patří detekce solárních panelů, kde se poškozený fotovoltaický článek typicky přehřívá, dále pak detekce poškození elektrického vedení, zjišťování tepelných ztrát ve stavebnictví a strojírenství, signalizace před vznikem požáru, nebo v lékařství pro měření teploty či detekci zánětů pod kůží. [13]

Infrakamera v oblasti bezkolizního pohybu autonomních prostředků by mohla mít využití například při vyhýbaní se lidem (viz. obrázek 11) či zvířatům nebo jejich následování.

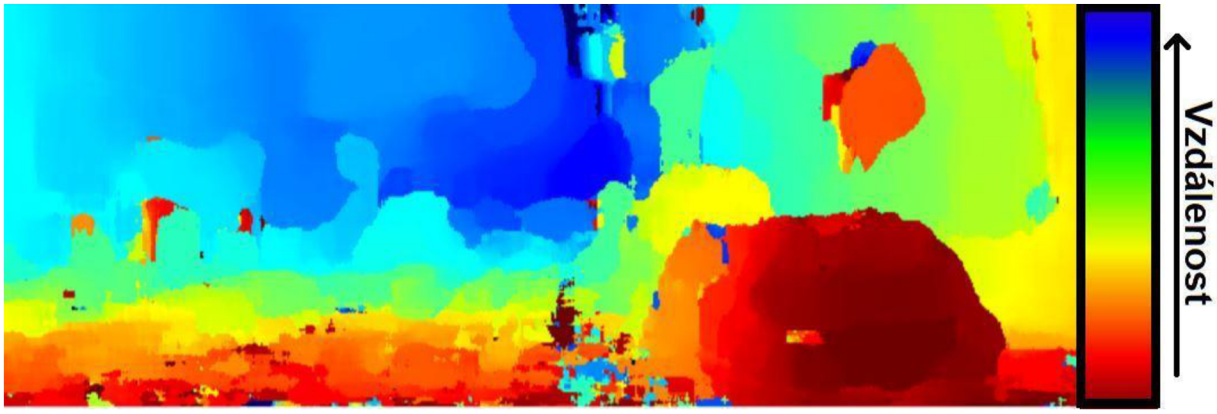


Obrázek 11 Detekce osob pomocí infrakamery [17]

3.1.2 Stereokamera

Způsob, jakým stereokamery fungují je ve velké míře podobný lidskému zraku. Stejně jako lidé mají dvě oči, stereokamery mají obvykle dva senzory. Mohou mít senzorů i více, ale pro jednodušší vysvětlení principu bude dále uvažováno, že mají senzory právě dva. Senzory u stereokamery jsou obvykle umístěny ve stejné horizontální úrovni, přičemž mají mezi sebou předem známou vzdálenost. Každý ze senzorů identifikuje obraz separátně. Na levém je celý

obraz posunut mírně vpravo a na pravém senzoru přesně naopak. Z obou obrazů, na kterých je zaznamenána ta samá scéna je zapotřebí identifikovat stejné pixely, které odpovídají stejnému reálnému objektu ve snímaném prostředí. To je obyčejně dost náročné, proto je možné využít toho, že obraz je na obou snímcích posunut pouze ve vodorovné ose, což umožňuje hledat pixely pouze v této ose. Nejprve je určen nějaký významný bod na snímku z levého senzoru, ideálně například hrana objektu. Poté se souřadnice tohoto bodu přenesou na snímek z pravého senzoru. V tu chvíli je jasné, že hledaný bod na pravém objektu bude vlevo od přenesených souřadnic. Pokud se najde stejný pixel, vyhodnotí se i okolí pixelu a když se i okolí shoduje, byl nalezen hledaný pixel. Rozdíl mezi umístěním stejné pixelu na levém a pravém obraze se nazývá disparita. Objekty, které jsou dále mají menší hodnotu disparity a ty co jsou blíže ji pak mají naopak vyšší. Z disparity a vzdálenosti mezi senzory lze určit hloubku obrazu, která umožňuje vytvořit 3D strukturu (viz. obrázek 12) neboli disparitní mapu snímaného prostředí [12].



Obrázek 12 Disparitní mapa snímaného prostředí [12]

Použitím stereokamer pro řízení autonomních prostředků lze získat stejně důležitá data jako použitím LiDARu, například vzdálenost, hloubku a následné generování mračna bodů, tj. 3D mapování prostředí. Obě technologie lze použít nejen k detekci objektů, ale také k jejich identifikaci při velkých rychlostech pohybu a při různých okolních podmínkách.

3.1.3 Digitální RGB kamera

Způsob zpracování videa touto kamerou je založen na jednoduchém principu. Barevná digitální kamera se skládá z objektivu, snímače a řídící elektroniky. Každý předmět, na který dopadá světlo ho část odrazí do prostoru. Toto světlo následně prochází objektivem kamery, ve kterém je soustava zrcadel, která ho usměrní na snímač. Snímačem obvykle bývá CMOS čip, který přeměňuje dopadající světlo na elektrický proud. Aby snímač dokázal vytvořit barevný obraz, musí na jeho fotobuňkách naneseny barevné filtry. Proud ze snímače je dále zpracován elektronikou na barevný digitální obraz [14].

Výstupem RGB kamery je barevný 2D obraz, ze kterého není možné vytvořit 3D mapu. Proto je samotná RGB kamera pro použití bezkolizního pohybu autonomních prostředků nevhodná. Z toho důvodu se v dnešní době častokrát k RGB kameře přidává hloubkový senzor,

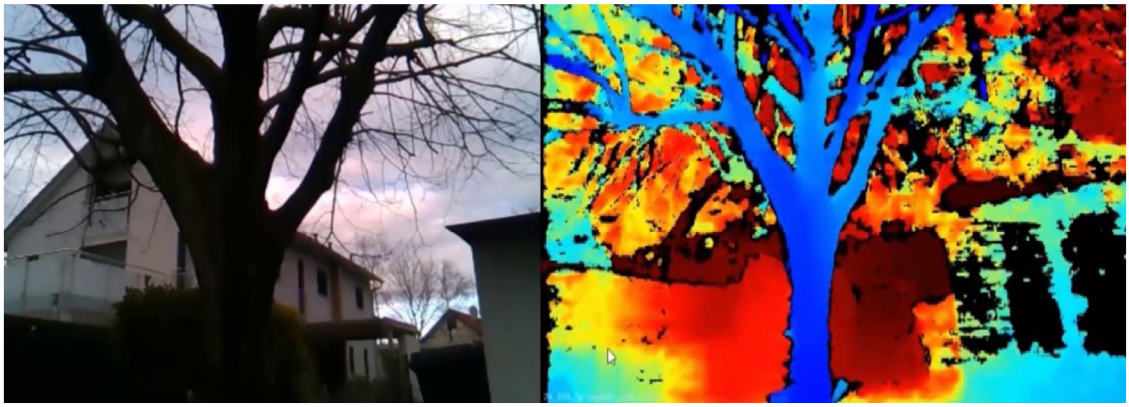
nebo bývá samotná RGB kamera součástí nějakého modulu, který obsahuje jiné snímače pro zjištění hloubky obrazu, například může obsahovat stereokameru.

3.1.4 Stereokamera Intel RealSense D455

Jedná se o modul od firmy Intel, obsahující několik senzorů (viz. obrázek 13) vhodných pro použití v aplikacích pro bezkolizní pohyb dronů či jiných robotických prostředků. Moduly od firmy Intel se od sebe liší v několika parametrech, přičemž základem všech modulů ze série D400 je stereokamera. Ta se skládá ze dvou čoček, tedy levé a pravé. Obě tyto čočky jsou identické. Intel Realsense D455 společně s několika dalšími moduly od této firmy mají i infračervený projektor, který emituje okolí laserem, což zlepšuje schopnosti stereokamery tak, že dokáže rozpoznat objekty i za zhoršených světelných podmínek. Laser, jenž projektor využívá pracuje na vlnové délce, kterou člověk nemůže vidět a zároveň mu není nijak nebezpečná. Dále moduly obsahují i klasickou digitální barevnou kameru. V čem se ale produkty ze série D400 mezi sebou liší je vzdálenost identifikace objektů. Tato vzdálenost se pohybuje v rozmezí 0,3 až 6 metrů. Liší se i v zorném poli, ve kterém dokážou senzory snímat okolí [15].

Obrázek 13 Intel RealSense D455 [15]

Ze stereokamery lze tedy vytvořit disparitní mapu (pravý záběr na obrázku 14), mračno bodů (třetí řádek na obrázku 28), klasický barevný obraz a z těchto dat je tedy možné získat 3D model okolí. To se perfektně hodí právě pro lokalizaci v neznámém prostředí, a to jak ve vnitřních, tak ve venkovních prostorách. V dokumentacích autopilotů PX4 a Ardupilot je možné jednoduše dohledat připojení těchto modulů s letovým řídícím kontrolérem Pixhawk, nastavení konfigurace i příklady použití.



Obrázek 14 Data získaná pomocí Intel RealSense D455

Modul Intel RealSense D455 je vhodným senzorickým vybavením pro bezkolizní řízení pohybu dronu v neznámém prostředí a z toho důvodu byl vybrán pro experiment v této práci.

3.2 LiDAR

„Light detection and ranging“ neboli LiDAR je stejně jako dva známé pojmy Sonar a Radar metoda měření vzdálenosti objektů. Všechny tyto technologie fungují na podobném principu. Vysílač vyšle signál, ten se odrazí od objektu a vrátí se zpět do přijímače. Je měřen čas mezi vyslaným a přijatým signálem, z toho je následně již jednoduché dopočítat danou vzdálenost objektu.

Hlavním rozdílem mezi LiDARem, Sonarem a Radarem je vysílaný signál. Radar vysílá elektromagnetické záření, konkrétně rádiové vlny o frekvenci desítek GHz, Sonar využívá ultrazvukových vln a LiDAR měří vzdálenost objektů pomocí laseru. LiDAR se skládá z několika hlavních částí, kterými jsou vysílač, příjmač, motor, optika, řídící elektronika.

Lasery, které se využívají v LiDARech mohou mít rozsah vlnových délek mezi 250nm až 2000nm. Aby nedošlo k poškození lidského zraku, tyto lasery typicky nepřevyšují výkon 1 W. Jelikož využívají světlo mohou mít problém při detekování objektů s černým povrchem, protože je světlo částečně pohlceno. Může nastat i situace kdy je zapotřebí skenovat například silnici, na které je voda. V tomto případě se jedná o téměř dokonale hladký povrch a žádný signál by se neodrazil zpět do příjmače. V takové situaci například při řízení autonomního vozidla by bylo zapotřebí využít dalšího senzoru, například kamery, která pracuje na jiném principu než LiDAR, a která dokáže silnici i s vrstvou vody detekovat.

LiDARy mohou fungovat na různých principech. Nejjednodušší metoda měření vzdálenosti objektů je princip TOF. Vysíláč emituje laserový paprsek, odrazí se od objektu a doputuje zpět do vysílače, přičemž je měřena doba letu paprsku. Ze vzorce (3.2.1) je pak snadné dopočítat měřenou vzdálenost.

$$d = \frac{c \cdot t}{2} \qquad (3.2.1)$$

Kde d představuje vzdálenost měřeného objektu, c reprezentuje rychlost světla a její hodnota je přibližně 300000 $m \cdot s^{-1}$, t udává čas, za který se světlo vrátí zpět do senzoru.

Další princip, který se u LiDARů využívá je měření fázového posuvu. Oproti metodě TOF laser nevysílá pulsy, naopak vysílá nepřetržitě. Následně je jednoduše porovnávána fáze vyslaného a přijatého signálu. Ze vzorce (3.2.2) je pak opět možné určit měřenou vzdálenost.

$$d = \frac{c \cdot \Delta\phi}{2\pi \cdot f} \qquad (3.2.2)$$

Kde d je vzdálenost měřeného objektu, c reprezentuje rychlost světla, $\Delta\phi$ udává fázový rozdíl přijatého a vyslaného signálu, f je frekvence signálu.

U triangulační metody se signál vyslaný přijmačem odrazí od překážky, doputuje na vstupní čočku optického přijmače pod určitým úhlem. Tento úhel se za čočkou vyhodnocuje jako vzdálenost světelného bodu na dopadové ploše přijmače tvořené CCD senzorem. Tato vzdálenost je pak přímo úměrná vzdálenosti měřeného objektu od snímače [9]. Právě triangulační metoda měření vzdálenosti snímaných objektů je využita u LiDARu RP LiDAR A3, který je použit v této práci.

LiDARy je možné rozdělit na ty umožňující měřit vzdálenosti ve dvou osách tedy 2D a na 3D, které mohou pracovat ve všech třech osách. 2D LiDARy vysílají pouze jeden paprsek a otáčejí se dokola. Jejich obvyklá maximální vzdálenost měření objektů je do 30 m. 3D LiDARy mají obvykle několik laserů a dokážou zaznamenat objekty až do vzdálenosti okolo 300 metrů.

V dnešní době existuje několik výrobců LiDARů pro využití při autonomní navigaci. Jedním z nich je RP LiDAR, jenž byl použit v tomto experimentu a je blíže rozebrán v následující části této práce. Dalším významným výrobcem LiDARů je Velodyne LiDAR. LiDARy od této značky se častokrát nasazují do reálných aplikací, protože oproti LiDARům RP LIDAR mají větší přesnost určení vzdálenosti, je možné pomocí nich měřit překážky až na stovky metrů a během jedné sekundy jsou schopné zaznamenat až 600000 bodů. Dalšími významnými výrobci LiDARů jsou firmy Luminar Technologies, Ouster, Aeva Technologies, AEye či Innoviz.

LiDARy se používají ve velkém množství aplikací, jimiž jsou výzkum atmosféry, autonomní navigace, v archeologii, kde je s jeho pomocí možné vytvářet digitální 3D modely archeologických lokalit. Dalším oborem, kde je možné se s LiDARem potkat je lesnictví. Zde pomáhá měřit výšku stromů, měřit hustotu porostu, určovat rozmanitost stromů [10].

3.2.1 RP LIDAR A3

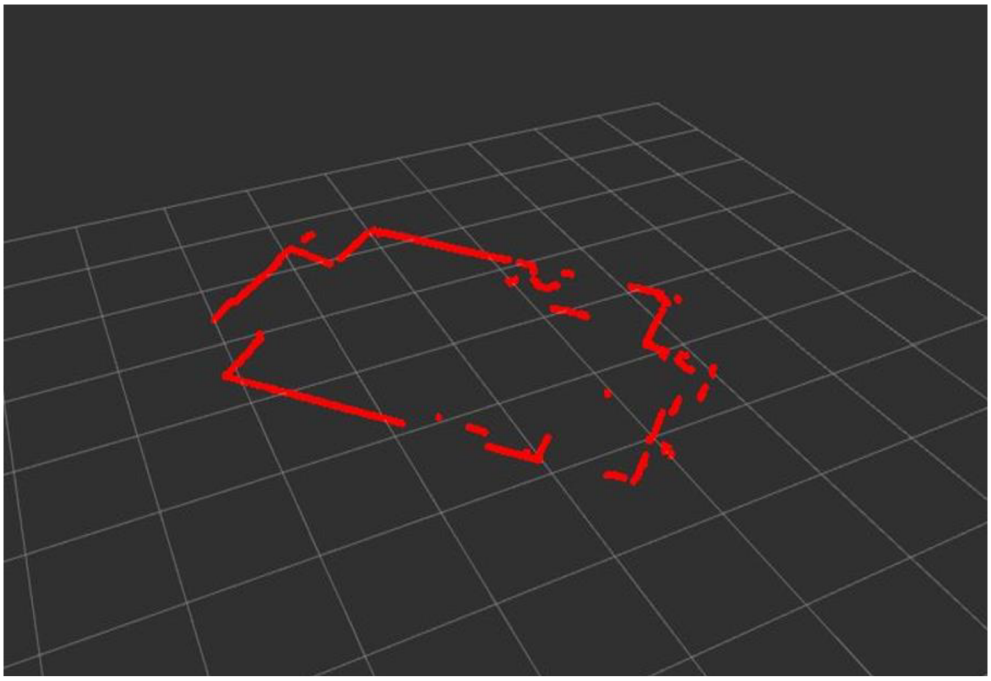
RP LIDAR A3 (viz. obrázek 15) pomocí rotačního systému dokáže snímat objekty ve 360 stupních kolem něho (viz. obrázek 16), a to ve vzdálenosti až 25 metrů. Umožňuje získat 16000 vzorků za sekundu i při vysokých rychlostech otáčení. Dokáže pracovat jak venku, tak ve vnitřních prostorách.

Obrázek 15 RP LIDAR A3 [11]

V porovnání s LiDARY A2 a A1 ze série RPLIDAR je lepší při detekování vzdálenějších objektů, bílých nebo černých povrchů a také umožňuje získat kvalitnější data kvůli možnosti

zaznamenávání více snímků za sekundu i při vyhodnocování objektů, které jsou na přímém slunečním světle. Modely S2 a S1 mají delší rozlišovací vzdálenost. S2 dokáže získávat až 32000 vzorků za sekundu ale má menší rozlišovací úhel. S1 je schopen vyhodnocovat objekty až do vzdálenosti 40 metrů, má vyšší rozlišovací úhel, ale oproti A3 ztrácí v počtu možných získaných vzorků za sekundu [11].

Model RPLIDAR A3 z těchto poznatků vychází jako ideální LiDAR pro využití při autonomní navigaci dronů, a právě z těchto důvodů je použit v experimentu této práce.



Obrázek 16 Zmapování prostoru pomocí RPLIDAR A3

4. REALIZACE METOD KOORDINACE LETU A ŘEŠENÍ KOLIZÍ

V této kapitole je podrobně rozebrán celý koncept řešení experimentu v reálném prostředí. V první části této kapitoly je diskutován problém řešení kolizí a je pro tento experiment navrhnut jednoduchý algoritmus pro detekování překážek na základě dat ze senzorů. Další část této kapitoly se zabývá hlavně nastavením řídícího počítače NVidia Jetson Nano a následným propojením senzorů s touto řídící deskou. Poslední část této kapitoly se věnuje primárně softwarové konfiguraci řídícího počítače. Pod tuto konfiguraci spadá vytvoření odkládacího prostoru pro dosažení lepších paměťových parametrů řídícího počítače. Taktéž je zde popsáno připojení obou senzorů použitých v tomto experimentu k řídícímu počítači společně s instalací nezbytných balíčků pro práci s těmito senzory. Závěr této kapitoly obsahuje informace o vlastním balíčku v softwaru ROS pro detekování překážek na základě získaných dat z obou senzorů.

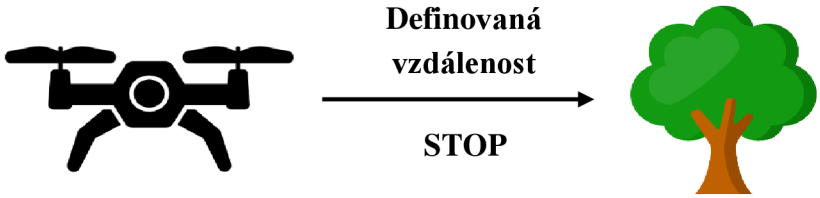
4.1 Řešení kolizí

Aby bylo možné přesně určovat vzdálenosti objektů a nedošlo tak ke kolizi dronu s překážkou, bylo využito dvou různých senzorů a data z nich získaná byla spojena do jednoho celku. Konkrétně byl použit RP LIDAR A3 a stereokamera Intel RealSense D400. Při detekci překážek pomocí LiDARu a stereokamery může nastat situace, že jeden z těchto senzorů má problém objekt zaznamenat. V tu chvíli je výhodné použití právě dvou senzorů pracujících na různých principech, protože druhý snímač by v tu chvíli neměl mít problém překážku zaznamenat.

Stereokamera byla v experimentu přidělána přímo k dronu a byla na dronu umístěna tak, aby sledovala scénu ve směru vpřed při letu dronu. Ze stereokamery byla získána disparitní mapa a mračno bodů z nichž je možné přepočítávat vzdálenosti objektů ve směru letu dronu.

LiDAR byl taktéž umístěn přímo na dronu. Oproti stereokameře je LiDAR schopen zaznamenávat překážky v celém rozmezí 360 stupňů kolem samotného dronu. Z LiDARu bylo získáno 2D mračno bodů, kde každý jednotlivý bod představuje zaznamenanou překážku. Z pozice těchto bodů je možné přepočítat vzdálenost jednotlivých překážek od samotného dronu,

Po získání dat z obou senzorů byl využit vlastní balíček v ROSu, u kterého při zaznamenání překážky dojde k varování pilota a případně by mohlo být dronu zakázáno další přiblížení se k objektu pod nějakou předem definovanou vzdálenost tak, aby nedošlo ke kolizi mezi dronem a samotnou překážkou. Zjednodušený případ je zobrazen na obrázku 17. Tato vzdálenost musí být dostatečně velká a v ideálním případě závislá na rychlosti pohybu dronu, protože kvůli vysoké náročnosti na všechny výpočty, není jednoduché zareagovat okamžitě.

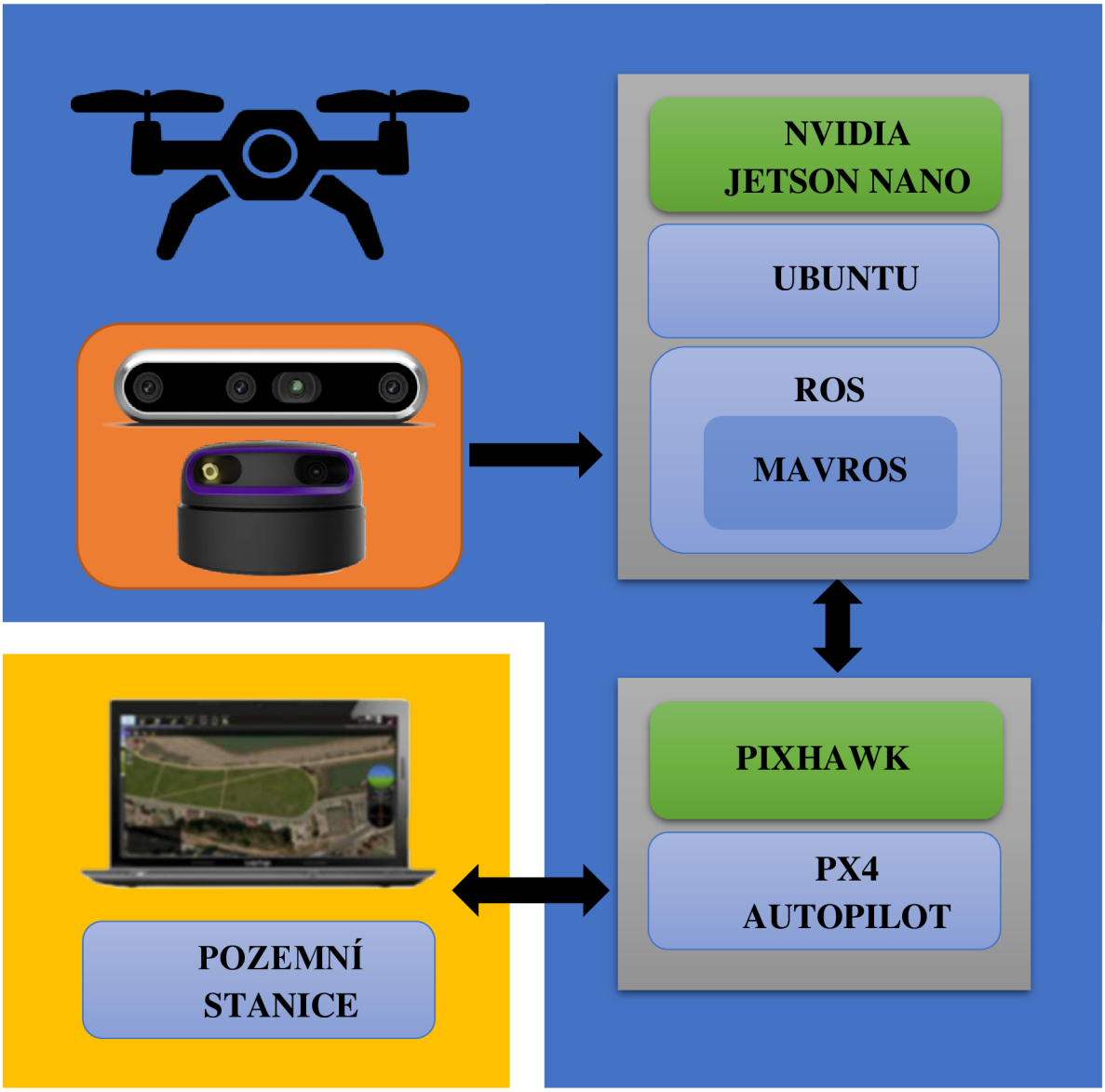


Obrázek 17 Bezkolizní let dronu

4.2 Propojení senzorů a řídícího počítače

Kompletní schéma zapojení hardwaru a softwaru je blokově znázorněno na obrázku 18. Hlavní prvek celého experimentu tvoří dron, který je dostatečně robustní, aby na jeho konstrukci mohl být připevněn potřebný hardware. Byl zvolen autopilot Pixhawk 4 a výkonná výpočetní jednotka NVIDIA Jetson Nano Developer Kit. Obě tyto jednotky mezi sebou sériově komunikují pomocí USB a jsou přidělány co nejblíže těžišti samotného dronu, aby neovlivňovaly samotný let. Zároveň na Pixhawk 4 se nachází šipka (viz. obrázek 20), která při letu dronu musí směřovat směrem vpřed.

Jako senzorické vybavení byl vybrán RP LIDAR A3 a kamerový modul Intel RealSense D455. Tyto použité senzory byly propojeny s Jetson Nano. Data shromážděná těmito senzory jsou v reálném čase odesílána do výpočetní jednotky Nvidia Jetson Nano k dalšímu zpracování. Pomocí těchto dat je dron schopen vnímat svoje okolí. Na výpočetní jednotce Jetson Nano vybavené operačním systémem Ubuntu 18.04 je nainstalován software ROS, na němž běží i potřebný algoritmus k docílení bezkolizního letu dronu. Při následném zakomponování řídícího počítače Pixhawk 4 by komunikoval s pozemní stanicí v tomto případě s PC. Komunikace by probíhala rádiově nebo prostřednictvím Wi-Fi. Na pozemní stanici je v tomto případě nainstalován software QGroundControl, v němž je možné sledovat aktuální data týkající se letu dronu a současně je možné do samotného letu dronu z pozemní stanice kdykoliv zasáhnou či naplánovat misi. Zároveň by bylo možné data pomocí MAVROS protokolu odesílat z Pixhawk 4 do Jetson Nano. O jejich překlad v řídícím počítači NVidia Jetson Nano se stará balíček v rosu MAVROS.



Obrázek 18 Blokové schéma zapojení experimentu

NVidia Jetson Nano Developer Kit

Jedná se o vestavěný systém na čipu, který je umístěn na desce (viz. obrázek 19) o rozměrech 69 x 100 mm a disponuje několika možnostmi připojení jako USB porty, gigabitový Ethernet nebo dva porty MIPI CSI-2 DPHY pro připojení kamery. Desku lze napájet buď 5 W nebo 10 W. K zajištění vyššího výkonu bylo využito napájení 10 W. Rozměry, hmotnost a omezení spotřeby umožňuje přímou montáž této desky na středně velký dron [25].

Na tomto vestavěném systému běží operační systém Ubuntu 18.04, kde je nainstalován i samotný ROS se všemi potřebnými balíčky pro získávání dat ze senzorů a jejich následné zpracování.

Obrázek 19 NVIDIA Jetson Nano Developer Kit [25]

Pixhawk 4

Konstrukci letového počítače PixHawk 4 lze pozorovat na obrázku 20. Dodává se s předinstalovaným nejnovějším firmwarem PX4. Součástí Pixhawk 4 je inerciální měřící jednotka (IMU), která pomocí kombinace gyroskopů a akcelerometrů podává informace o zrychlení a orientaci v prostoru. Kromě IMU jsou v Pixhawk 4 také umístěny magnetometr a barometr. Kde magnetometr funguje jako kompas a barometr dokáže určovat tlak vzduchu. Všechny tyto zmíněné palubní senzory, fungují jako zpětná vazba pro řídící a navigační systém.

Pro nastavení samotného Pixhawk 4 je nutné mít naistalovaný software QGroundControl, spojit řídící jednotku pomocí USB s Pixhawk 4, který je softwarem automaticky rozpoznán. V prostředí PX4 je poté zapotřebí udělat několik povinných kalibrací jakožto kalibrace kompasu, akcelerometru a podobně.

Kromě povinných kalibrací si uživatel může pomocí sběrnic I2C, SPI nebo CAN připojit další vlastní hardware. Pixhawk 4 má taktéž PWM výstupy, které lze použít například pro ovládání BLDC motorů nebo servopohonů.

Obrázek 20 Pixhawk 4 [27]

4.3 Konfigurace řídícího počítače pro fúzi senzorů

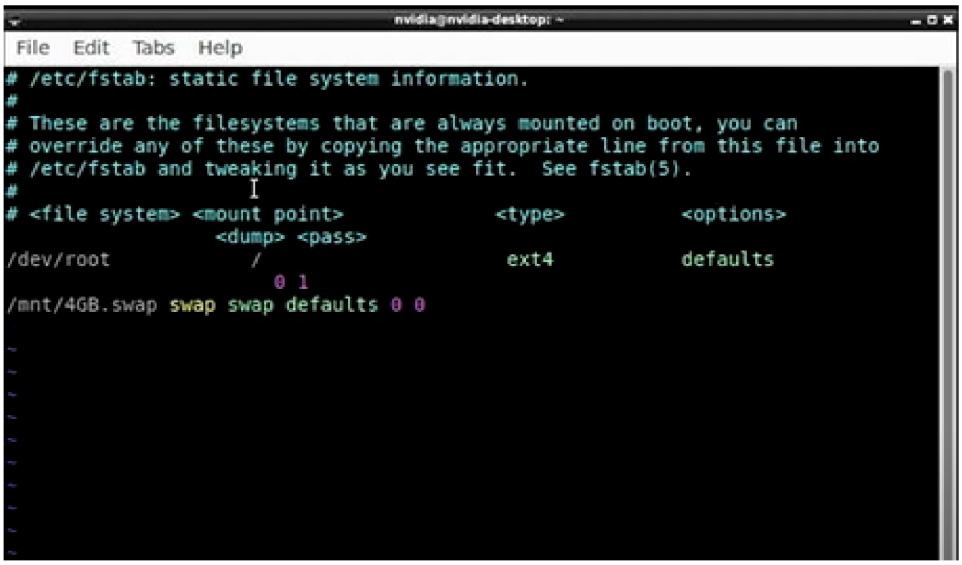
Spouštění této řídící jednotky je možné prostřednictvím klasického monitoru. Druhou možností je spouštění Jetsonu v tzv. „headless modu“. To zjednodušeně znamená ovládání této jednotky z jiného počítače například využitím SSH klientu PuTTy. V případě tohoto experimentu pro prvotní spuštění a nastavení jednotky Nvidia Jetson Nano bylo využito klasických periférii jako monitor, myš a klávesnice. Pro napájení byl využit napájecí adaptér s Micro-USB konektorem. Pro pozdější instalaci všech náležitostí na řídící jednotce bylo třeba mít desku připojenou k internetu. Při konfiguraci tohoto řídícího počítače pro fúzi senzorů byla deska Jetson Nano připojena k síti pomocí klasického ethernet kabelu. Dalším způsobem by mohlo být připojení k WiFi síťi a to například pomocí bezdrátového modulu. Po vložení SD karty do slotu modulu Jetson Nano, na kterém byl nahraný image s operačním systémem Linux, bylo možné řídící jednotku připojit k napájení a tím se sama spustila. Následně se na obrazovce monitoru zobrazilo okno s konfigurací systému. Zde bylo zapotřebí projít velice intuitivním procesem nastavení několika nezbytných náležitostí. Po dokončení konfigurace byl systém připraven.

Vytvoření odkládacího prostoru

Jelikož při práci s řídící deskou Nvidia Jetson Nano, hlavně při instalaci a sestavování balíčků častokrát dochází k vyčerpání fyzické RAM paměti, proto bylo více než vhodné si vytvořit odkládací prostor. Tento odkládací prostor neboli swap file je místo na disku, které se používá právě když je v systému Linux zaplněna paměť RAM. Pro vytvoření swap file o velikosti 4GB bylo zapotřebí zapsat následující příkazy do termínálu.

```
$ sudo fallocate -l 4G /mnt/4GB.swap
$ sudo chmod 600 /mnt/4GB.swap
$ sudo mkswap /mnt/4GB.swap
```

Pro nastavení tohoto swap file tak, aby se nemusel vytvářet znovu při každém spuštění systému byl otevřen konfigurační soubor /etc/fstab. Konfigurační soubor bylo nutné upravit například využitím textového editoru vi tak, jak je vidět na obrázku 21. Zápis do toho souboru byl umožněn po stisku znaku i (insert) na klávesnici. Dané změny bylo třeba uložit a poté restartovat systém.

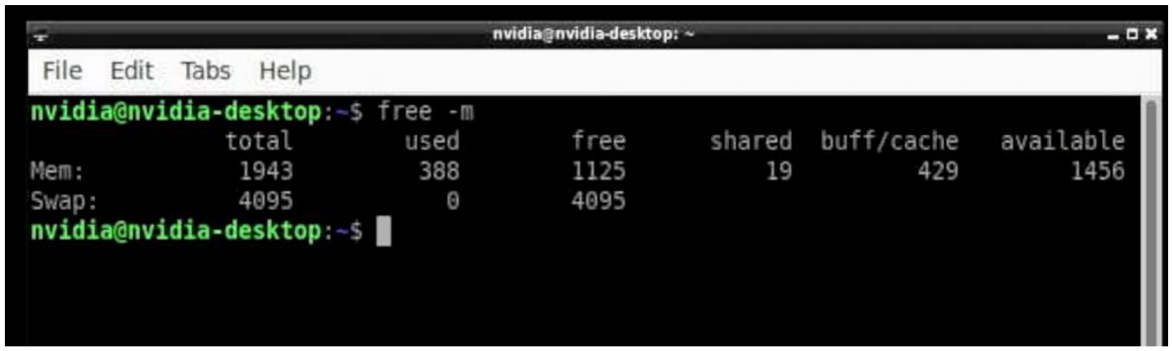


Obrázek 21 Konfigurační soubor pro nastavení swap file

Po opětovném spuštění systému bylo třeba ověřit, zda byl swap file opravdu vytvořen, a to pomocí zapsání následujícího příkazu do terminálu.

```
$ free -m
```

V terminálu se vypsalo několik řádků (viz. obrázek 22) s údaji o RAM paměti a swap file. Tato data jsou udávána v MB. Jelikož velikost swap paměti odpovídá nastavené velikosti, bylo vše nastaveno správně a systém je připravený na používání.



Obrázek 22 Aktuální stav používané paměti

4.3.1 Instalace Robotic Operating System

Aktuální verze ROSu je pro distribuci Ubuntu 18.04 ROS Melodic a pro Ubuntu 20.04 je to verze ROS Noetic. Řídící deska NVidia Jetson Nano podporuje pouze verzi Ubuntu 18.04, proto byla na tuto desku nainstalována verze ROSu Melodic.

Nejprve bylo nutné přidat repozitář do Linuxu, ze kterého bylo možné ROS stáhnout

```
$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

Přidání klíče do systému pro zabezpečení komunikace s repozitářem

```
$ sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

Načtení nově přidaných dat

```
$ sudo apt update
```
Instalace samotné ROSu podle zvolené verze, v tomto případě verze Melodic.
```
$ sudo apt install ros-melodic-desktop-full
```
Na závěr bylo provedeno nastavení ROSu pomocí následujících příkazů tak, aby byl spustitelný kdykoliv při otevření nového terminálu.
```
$ echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
$ source ~/.bashrc
```
Dalším důležitým krokem bylo vytvoření vlastní složky, kde byly následně instalovány všechny používané balíčky v ROSu. Pro tento přístup bylo užitečné využít pracovního prostoru Catkin, což je pracovní prostor, který sdružuje balíčky v ROSu dohromady.
```
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/
```
Aby nebylo zapotřebí při každém spuštění nového terminálu nastavovat cestu k bash souboru, bylo to nastaveno bylo to nastaveno pevně a to pomocí následujících kroků. Nejpve bylo nutné otevřít textový soubor bashrc a to například pomocí programu gedit. Instalace textového editoru gedit byla spuštěna následujícím příkazem.
```
$ sudo apt-get install gedit
```
Následně bylo nutné otevřít textový soubor bashrc.
```
$ gedit ~/.bashrc
```
Textový soubor byl upraven tak, aby na svém konci obsahovat následující dva řádky.
```
source /opt/ros/melodic/setup.bash
source ~/catkin_ws/devel/setup.bash
```
Na závěr zbývalo už jen nastavit cestu k bash souboru ve vytvořeném pracovním prostoru, a to napsáním následujícího příkazu do terminálu
```
$ source ~/catkin_ws/devel/setup.bash
```

4.3.2 Připojení LiDARu a instalace potřebných balíčků

V experimentu byl použit model RPLIDAR A3. Pro práci a nastavení tohoto LiDARu bylo nutné ho připojit k řídícímu počítači pomocí USB. Po jeho připojení k desce bylo vhodné v terminálu zkontrolovat, zda je opravdu připojený, a to lze učinit pomocí následujícího příkazu.
```
$ ls -l /dev | grep ttyUSB
```
Poté bylo třeba LiDARu nastavit potřebná povolení.
```
$ sudo chmod 666 /dev/ttyUSB0
```
Tímto byl LiDAR připraven ke komunikaci s řídící deskou.

Instalace balíčků rplidar_ros a hector_slam

Balíček rplidar_ros funguje jako driver pro RPLIDAR A3. Pomocí SDK čte skenovaná data z LiDARu a převádí je na zprávy v ROSu.

Balíček hector_slam je algoritmus, který využívá data z laserového skenování pomocí LiDARu k vytvoření 2D mapy skenovaného prostředí. Výhodou tohoto balíčku oproti jiným technikám SLAM je, že ke své práci potřebuje pouze data z laserového skenování.

Pro instalaci obou balíčků bylo zapotřebí si v terminálu otevřít vytvořený pracovní prostor catkin workspace a naklonovat pomocí gitu samotné balíčky.
```
$ cd catkin_ws/src
```

V této složce se klonují všechny používané balíčky. V tomto případě bylo zapotřebí pomocí gitu naklonovat balíček rplidar_ros a hector_slam.

```
$ sudo git clone https://github.com/Slamtec/rplidar_ros.git
$ sudo git clone https://github.com/tu-darmstadt-ros-pkg/hector_slam.git
```

Po naklonování nových balíčků bylo nutné se vrátit zpět o složku níže a pomocí programu Catkin sestavit celý pracovní prostor i s novými „balíčky".

```
$ cd ~/catkin_ws/
$ catkin_make
```

Tyto balíčky obsahují několik předpřipravených spustitelných souborů (launch files), které si sami zavolají a spustí jednotlivé nody v ROSu. Příkladem je tutorial launch file. Pro jeho start bylo nutné mít připojený LiDAR, pomocí terminálu vstoupit do pracovního prostoru s nainstalovanými balíčky a spustit launch file rplidar_a3, který následně zapnul samotný LiDAR.

```
$ cd ~/catkin_ws/
$ roslaunch rplidar_ros rplidar_a3.launch
$ roslaunch hector_slam_launch tutorial.launch
```

Zavoláním těchto příkazů se odstartoval LiDAR a spustilo simulační prostředí RViz. V tomto simulačním prostředí byly zobrazovány jednotlivé topiky. Pro přidání topiku, který představuje mračno bodů v 2D souřadnicích, které reprezentují zaznamenané překážky, bylo nutné vlevé dolní části kliknout na tlačítko „add", vybrat kolonku „by topic" a zde v části „/scan" zvolit „LaserScan". Obdobně pak bylo třeba přidat další topiky, které byly v simulačním prostředí RViz zobrazovány.

4.3.3 Připojení stereokamery a instalace potřebných balíčků

Stereokamera použitá v tomto experimentu je Intel RealSense D455. Pro její připojení k řídícímu počítači bylo opět využito klasického USB.

Instalace RealSense SDK

Jedná se o multiplatformní knihovnu pro stereokamery Intel RealSense. Poskytuje vnitřní a vnější kalibrační informace, hloubkové a barevné datové toky, syntetické datové toky, jako je mračno bodů, hloubka scény zarovnaná na barvu apod. Pro instalaci této knihovny bylo zapotřebí odpojit připojenou kameru Intel RealSense a do terminálu zapsat následující příkazy.

```
$ sudo apt-get update && sudo apt-get upgrade && sudo apt-get dist-upgrade
$ git clone https://github.com/IntelRealSense/librealsense.git
$ cd ~/librealsense
$ sudo apt-get install git libssl-dev libusb-1.0-0-dev pkg-config libgtk-3-
dev
$ sudo apt-get install libglfw3-dev libgl1-mesa-dev libglu1-mesa-dev at
$ ./scripts/setup_udev_rules.sh
$ ./scripts/patch-realsense-ubuntu-lts.sh
$ mkdir build && cd build
$ cmake / -DBUILD_EXAMPLES=true
$ sudo make uninstall && make clean && make && sudo make install
```

Instalace ROS Wrapperu pro zařízení Intel RealSense

Jedná se o balíčky pro kamery RealSense série D400, kameru SR300 a pro modul T265. Tyto balíčky umožňují komunikaci kamer se softwarem ROS. Pro jejich instalaci bylo nutné do terminálu vložit následující příkazy.

```
$ cd ~/catkin_ws/src
$ git clone https://github.com/IntelRealSense/realsense-ros.git
$ cd realsense-ros
$ git checkout `git tag | sort -V | grep -P "^2.\d+\.\d+" | tail -1`
$ cd ~/catkin_ws
$ catkin_make
```

Instalace rozšířených balíčků pro SLAM.

```
$ sudo apt-get install ros-melodic-imu-filter-madgwick
$ sudo apt-get install ros-melodic-rtabmap-ros
$ sudo apt-get install ros-melodic-robot-localization
```

Stejně jako u balíčku hector_slam je zde připraveno několik launch files, které spustí jednotlivé nody v ROSu. Opět bylo zapotřebí být v pracovním prostoru, kde jsou naistalovány všechny balíčky. Následně stačilo spustit zvolený launch file.

```
$ cd catkin_ws
$ roslaunch realsense2_camera opensource_tracking.launch
```

Stejně jako u balíčků pro LiDAR se spustilo simulační prostředí RViz, ve kterém bylo třeba si v levé části vybrat topiky, které byly v simulátoru zobrazovány.

4.3.4 Balíček pro detekci překážek

V tomto experimentu byl také vytvořen vlastní balíček v ROSu, který dokáže vyhodnotit překážky v předem definované vzdálenosti od dronu. V řídícím počítači Nvidia Jetson Nano byl využit již zmiňovaný pracovní prostor catkin workspace, ve kterém byl umístěn tento balíček. Instalace balíčku byla provedena pomocí následujících příkazů.

```
$ cd ~/catkin_ws/src
$ catkin_create_pkg obstacle_detection rospy
```

Následně bylo zapotřebí v balíčku obstacle_detection vytvořit novou složku launch, ve které se nachází launch file, který poté sloužil pro spouštění kódu pro detekci překážek. V tomto vytvořeném launch file bylo zapotřebí vložit odkaz na kód (ROS node). Tento kód byl napsán v programovacím jazyce Python a byl umístěn ve složce catkin_ws/src/obstacle_detection/src. ROS node v tomto balíčku se chová jako subscriber, tedy poslouchá topiky ze stereokamery a z LiDARu a tato data zpracovává tak, aby došlo k detekci překážek.

5. EXPERIMENT V REÁLNÉM PROSTŘEDÍ

V této kapitole jsou diskutovány dosažené výsledky získaných dat ze senzorů umístěných na samotném dronu (viz. obrázek 23). V první části, která je zde rozebrána, jsou data získaná z LiDARu RP LiDAR A3. Jedná se o mračno bodů v 2D prostoru, které je vhodné pro detekci objektů v jedné rovině a zároveň může být dále zpracováváno například využitím SLAM algoritmu.

Další částí, která je v této kapitole probírána jsou data získaná ze senzorického modulu Intel RealSense D455. Zpracováním dat z toho snímače vzniklo klasické barevné video, disparitní mapa snímaného prostředí a 3D mračno bodů.

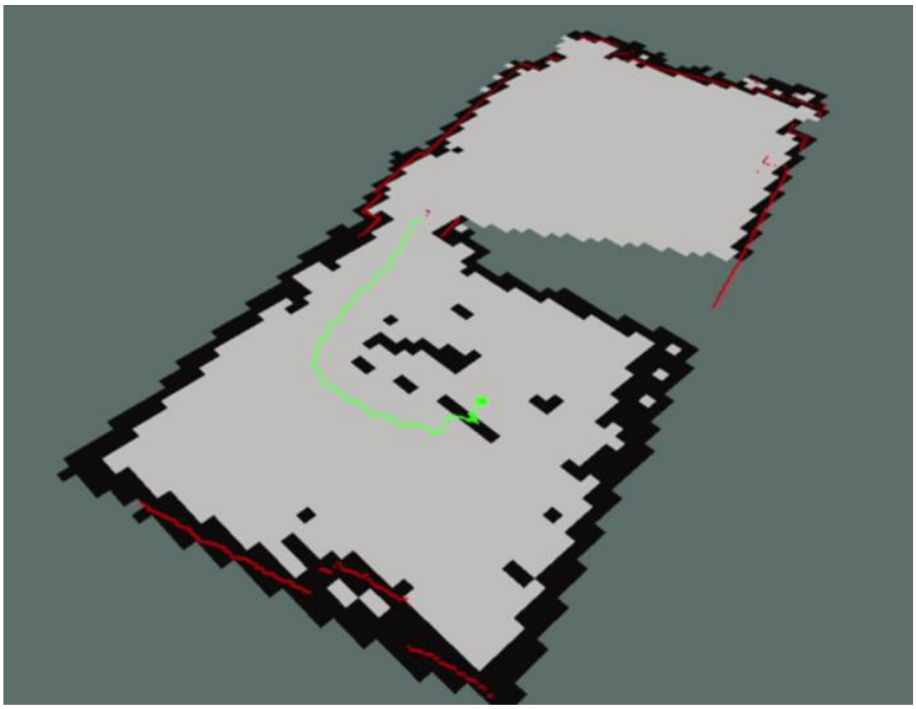
Na závěr této kapitoly je popsáno celkové výsledné zpracování dat z obou senzorů, kde jsou tato data spojena do jednoho celku tak, aby je bylo možné efektivně využít při dalším využití například pro bezkolizní let v neznámém prostředí.

Obrázek 23 Dron využitý v experimentu

5.1 Zpracování dat z LiDARu

Pro zpracování a reprezentaci dat získaných LiDARem RP LIDAR A3 bylo využito balíčků v ROSu rplidar_ros a hector_slam. Pomocí těchto balíčků lze transformovat data z LiDARu do 2D mračna bodů (viz. červené body na obrázku 24). Každý z těchto bodů je určen souřadnicemi v 2D prostoru, z čehož je následně možné přiřadit jednotlivému bodu, který představuje objekt v prostoru, vzdálenost od samotného dronu. Z těchto údajů lze následně v reálném čase efektivně provádět lokalizaci dronu a vytvářet mapu prostředí (SLAM). V rámci této práce byl SLAM algoritmus testován jak ve vnitřních, tak ve venkovních prostorech. Při experimentu v uzavřeném prostoru bylo dosaženo relativně dobrých výsledků (viz. obrázek 24), tedy mapování místnosti se souběžnou lokalizací se shodovalo s realitou. Oproti tomu SLAM algoritmus byl téměř nepoužitelný ve venkovních prostorech. Docházelo k tomu, že algoritmus

kvůli rychlým pohybům, nevhodnému terénu a přímému slunci ztrácel údaje o poloze dronu, a proto mapování a lokalizace neprobíhaly tak, jak by měly. Zlepšení by určitě přineslo doplnění údajů o přesné odometrii (změna pozice a orientace), například z IMU jednotky, kterou obsahuje senzorický modul Intel RealSense D455 či IMU jednotky v Pixhawk 4.



Obrázek 24 Zpracovaná data získaná z LiDARu

5.2 Data ze senzorického modulu Intel RealSense D455

Balíček realsense-ros byl použit pro zpracování a reprezentaci dat získaných ze senzorického modulu Intel RealSense D455. Tím nejdůležitějším, co bylo vytvořeno pomocí tohoto balíčku je 3D mračno bodů, kde každý bod v prostoru může být zobrazován i reálnými barvami, protože lze spojit data z RGB kamery společně s daty ze stereokamery do jednoho celku. Opět každý bod v tomto mračnu bodů je definován souřadnicemi v prostoru, tudíž lze jednoduše vypočítat vzdálenost jednotlivého bodu od samotného dronu. Dalším důležitým výsledkem, který vznikl, je obraz v šedých barvách neboli disparitní mapa (druhý řádek na obrázku 28), kde se odstín šedé barvy zaznamenané scény mění se vzdáleností. Tedy nebližší objekty mají nejtmavší šedou barvu a ty nejvzdálenější zase naopak světlejší šedou barvu. Posledním, co bylo využito z tohoto senzorického modulu je klasické video z RGB kamery.

V tomto experimentu bylo otestováno i vytváření 3D mapy z jednotlivých mračen bodů. Jelikož je tento algoritmus velmi náročný na paměť, není tento přístup při tomto použitém hardwaru moc vhodný pro využití na samotném dronu. Pro dosažení kvalitního mapování 3D prostoru pomocí tohoto balíčku by mohlo značně pomoci snížení zaznamenávaného rozlišení ze stereokamery, snížení rychlosti pohybu dronu tak, aby nezaznamenával nové prostředí tak často a ideálně značné navýšení velikosti RAM paměti řídícího počítače.

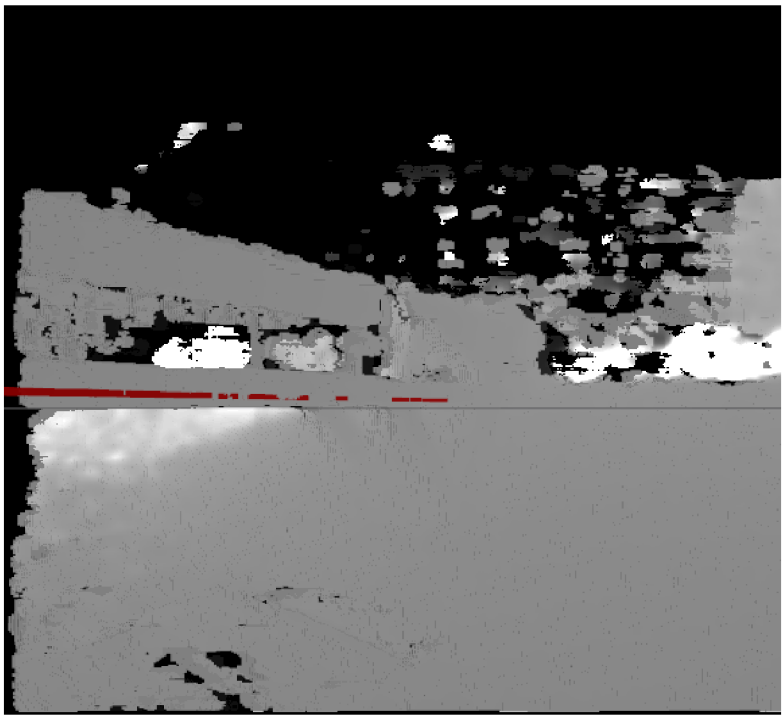
5.3 Senzorická fúze

Při testu senzorické fúze na dronu v reálných podmínkách bylo získáno několik vhodných dat pro plánování pohybu dronu v neznámém prostředí. Všechna tato data, respektive jednotlivé topiky byly ukládány do ROS bagu, ze kterého bylo následně možné výsledky znovu přehrát nebo nějakým způsobem zpracovat či zobrazit v simulačním prostředí RViz stejně, jako tomu bylo v tomto experimentu.

Obrázek 25 představuje obraz z klasické barevné kamery nahraný v reálném prostředí při experimentu fúze senzorů.

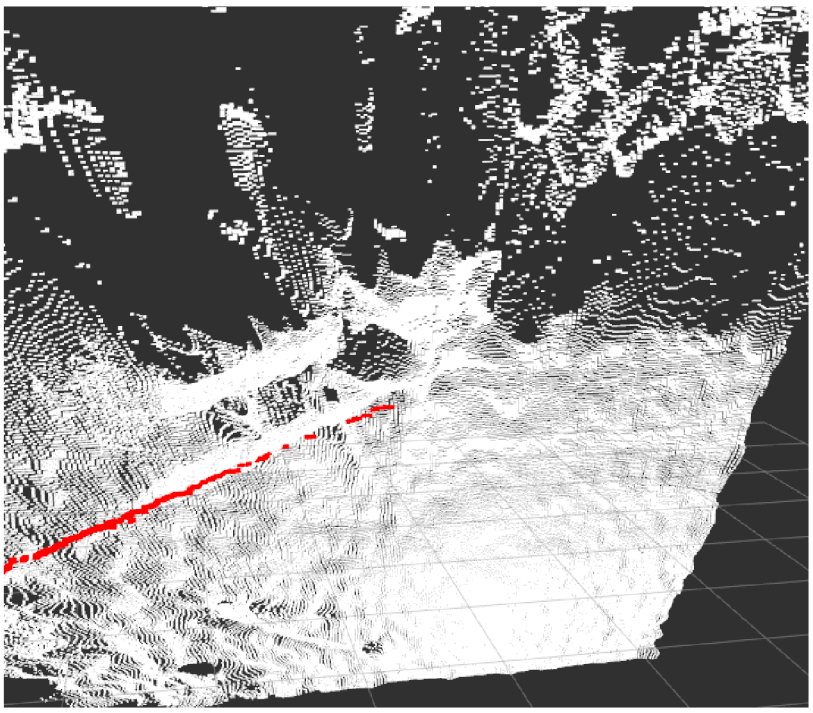
Obrázek 25 Barevný obraz získaný senzorickou fúzí

Další získaná a zpracovaná data je možné vidět na obrázku 26. Jedná se o stejnou reálnou scénu jako na obrázku 25. Konkrétně je možné pozorovat disparitní mapu vytvořenou z dat ze stereokamery. Odstíny šedé zde odpovídají vzdálenosti překážky. Tedy nejsvětlejší je nejvíce vzdálená překážka a naopak šedá, která je nejvíce tmavá, představuje nejbližší překážky. Černá barva znamená, že zde kamera překážku nezaznamenala. Obrázek 26 taktéž obsahuje 2D mračno bodů vytvořené z dat z LiDARu. Tyto body jsou na obrázku 26 označeny červenou barvou. Data z LiDARu se v tomto případě velmi dobře shodují s daty ze stereokamery.



Obrázek 26 Disparitní mapa sloučená s mračnem bodů z LiDARu

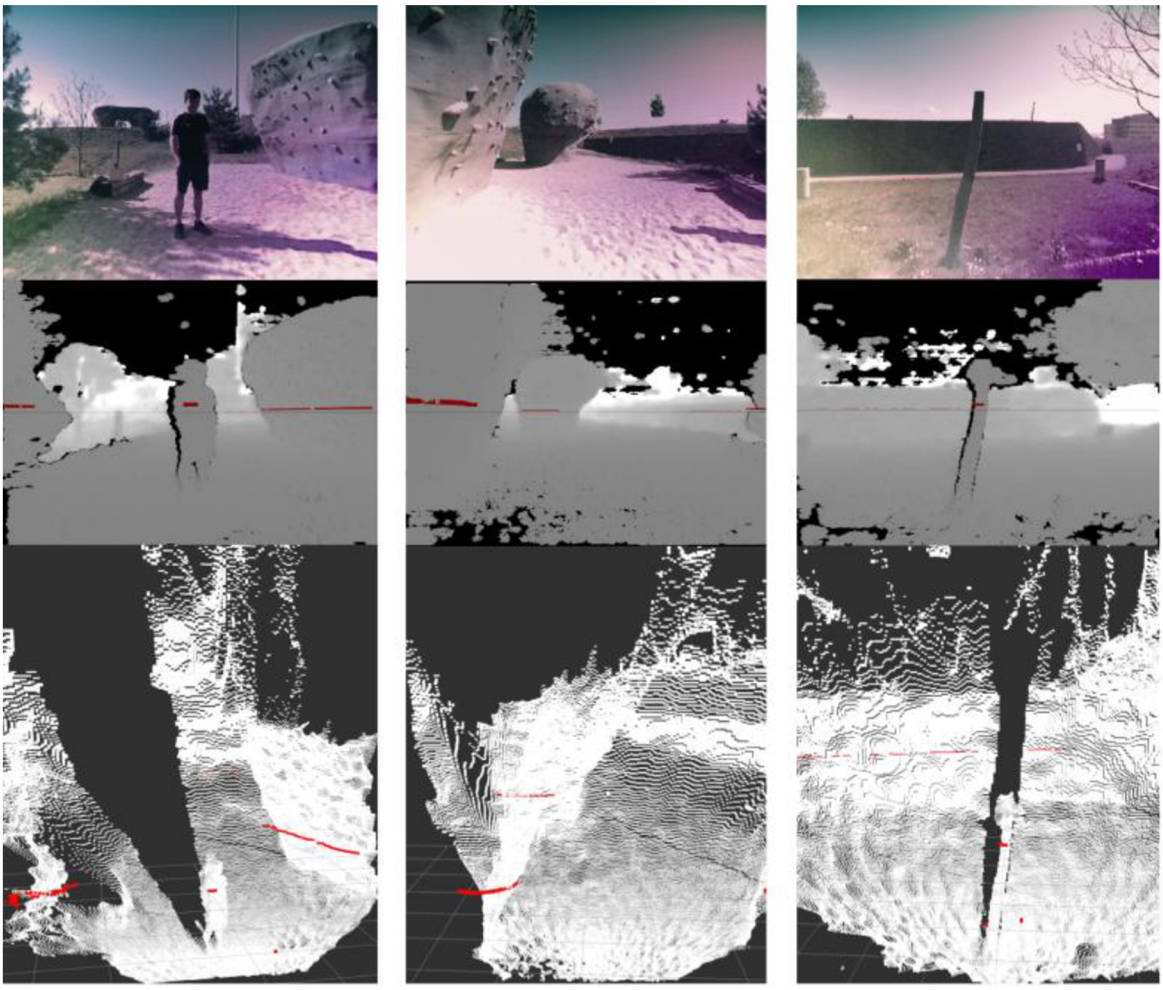
Obrázek 27 zaznamenává opět stejnou scénu jako na obrázku 25. Bílé body představují 3D mračno bodů získané ze stereokamery. Tyto body udávají reálné objekty v prostoru. Taktéž je na obrázku 27 opět červeně zobrazeno 2D mračno bodů získané z LiDARu. Data ze stereokamery velmi dobře korespondují s daty z LiDARu.



Obrázek 27 Spojená mračna bodů z LiDARu a stereokamery

Obrázek 28 ukazuje opět senzorickou fúzi v reálném prostředí. Tentokrát byly snímány tři různé scény. Na první scéně (sloupec vlevo) je vidět, že oba senzory správně detekovaly osobu jako překážku. Scéna zobrazená v prostředním sloupci představuje detekci lezeckých stěn. Opět oba senzory tyto stěny zaznamenaly. Na poslední scéně je vidět sloup se stěnou v pozadí, které byli taktéž pomocí obou senzorů správně detekovány.

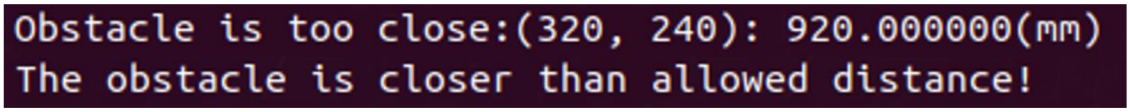
Data získaná z obou senzorů jsou vhodná pro použití při plánování pohybu dronu v neznámém prostředí. Častokrát však nastává situace, že jeden ze senzorů kvůli jeho fyzikálnímu principu nemá dostatečně velkou snímací vzdálenost, ztratí svou kvalitu snímání okolí nebo vůbec překážky nezaznamenává. Takových situací, kdy RP LIDAR A3 nesprávně snímá okolí, je hned několik. Příkladem je detekce překážky, která je průhledná, snímání prostředí, kde se víří prach, při dešti či padání sněhu, detekce černých objektů. Oproti tomu senzorický modul Intel RealSense D455 má problémy při snímání scény, která je přesvícená, funguje na kratší vzdálenost, sleduje scénu pouze před sebou či jeho RGB obraz samozřejmě není použitelný ve tmě.



a) b) c)

Obrázek 28 Senzorická fúze v reálném prostředí

V reálném experimentu fúze senzorů byl taktéž otestován vlastní balíček v ROSu obstacle_detection. Po spustění launch file probíhala detekce překážek, a to na základě dat z obou použitých senzorů. Při detekovaní překážky v reálném experimentu byly do terminálu vypsány hlášky zobrazené na obrázku 29. Tento koncept v reálných podmínkách dokázal efektivně detekovat stěnu, člověka či strom.



Obrázek 29 Varování pilota při detekci překážky

Tyto informace je možné v rámci ROSu dále zpracovávat a například je předat letovému počítači PixHawk 4, který by na základě této informace provedl zastavení letu dronu či jakoukoliv další předem definovanou operaci.

6 ZÁVĚR

Jedním z úkolů bakalářské práce bylo seznámit se s navigací dronů v neznámém prostředí. Tato problematika si sebou nese i legislativní pravidla, která byla vztažena na území České republiky a byla popsána v rámci této práce. K docílení navigace v neznámém prostředí je zapotřebí senzorického vybavení. Žádný senzor není ideální pro komplexní využití ve všech podmínkách. V rámci této práce byly zmíněny a popsány senzory vhodné pro autonomní řízení a rovněž byly diskutovány všechny jejich výhody a nevýhody. Pro experiment fúze senzorů pro plánování pohybu dronu, byly vybrány RP LIDAR A3 a senzorický modul Intel RealSense D455 jako potřebné senzory pro detekování překážek v okolí dronu. Oba z těchto zmíněných snímačů disponují svými výhodami i nevýhodami. Častokrát tedy může nastat situace, kdy jeden ze senzorů není vhodný pro aktuální zaznamenávání okolí. Z tohoto důvodu využitím softwaru ROS byla data z obou použitých senzorů sloučena do jednoho celku tak, aby se předešlo možným chybám detekce objektů v důsledku nesprávného měření kvůli fyzikálnímu principu jednotlivého snímače. Celý postup instalace potřebného softwaru, jeho nastavení, připojení veškerého hardwarového vybavení je taktéž v této práci důkladně popsán. Výsledná senzorická fúze obsahuje sjednocení 2D mračna bodů z LiDARu s 3D mračnem bodů ze stereokamery. Také byla z dat senzorického modulu Intel RealSense D455 získána disparitní mapa snímaného prostředí spolu s klasickým barevným obrazem. Mračna bodů vznikla primárně pro využití při bezkolizním pohybu dronu. Barevný obraz z RGB kamery může být jednoduše využit pro rozpoznávání objektů či detekci osob. V této práci byl taktéž vytvořen vlastní balíček v softwaru ROS, který na základě dat získaných ze senzorů vyhodnocuje překážky v okolí dronu a o této překážce dokáže varovat pilota. Tyto informace o překážkách je možné následně prostřednictvím softwaru ROS dále zpracovávat a například je posílat letovému počítači PixHawk 4, který by na základě těchto informací buď dokázal zastavit dron v bezpečné vzdálenosti od dané překážky nebo využitím pokročilých algoritmů by dokázal povelovat dron tak, aby se dané překážce efektivně vyhnul.

Zahrnutí letového počítače PixHawk 4 do tohoto systému by bylo přímé navázání na práci v tomto experimentu, tato problematika již není součástí zadání této práce. Bylo by možné přímo využít data získaná v tomto experimentu a dále je pomocí balíčků v ROSu zpracovávat tak, aby docházelo k lokalizaci dronu a mapování prostředí. Při získání mapy okolí dronu by bylo možné aplikovat další balíčky v ROSu pro řízení dronu pomocí PixHawk 4 tak, aby se autonomně vyhýbal překážkám.

LITERATURA

[1] Portál dronpro.cz [online]. [cit. 29.10.2021]. Dostupné z: https://dronpro.cz/existuje-zcela-autonomni-dron-poznejte-5-stupnu-letecke-autonomie#level-5-plna-automatizace

[2] Bezpilotní letadla - Úřad pro civilní letectví. Úřad pro civilní letectví - Bezpečně a s nadhledem [online]. [cit. 29.10.2021]. Dostupné z: https://www.caa.cz/provoz/bezpilotni-letadla

[3] Lentin J., Mastering ROS for Robotics Programming - Design, build and simulate complex robots using Robot Operating System and master its out-of-the-box functionalities, Packt Publishing, 2015, ISBN 978-1783551798 [cit. 29.12.2021]

[4] NOORI, Farzan M., David PORTUGAL, Rui P. ROCHA a Micael S. COUCEIRO. On 3D simulators for multi-robot systems in ROS: MORSE or Gazebo? In: 2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR) [online]. IEEE, 2017, 2017, s. 19-24 [cit. 2021-11-06]. ISBN 978-1-5386-3923-8. Dostupné z: doi:10.1109/SSRR.2017.8088134

[5] TAKAYA, Kenta, Toshinori ASAI, Valeri KROUMOV a Florentin SMARANDACHE. Simulation environment for mobile robots testing using ROS and Gazebo. In: 2016 20th International Conference on System Theory, Control and Computing (ICSTCC) [online]. IEEE, 2016, 2016, s. 96-101 [cit. 2021-11-06]. ISBN 978-1-5090-2720-0. Dostupné z: doi:10.1109/ICSTCC.2016.7790647

[6] Introduction · MAVLink Developer Guide. Choose a language · MAVLink Developer Guide [online]. [cit. 29.12.2021] Dostupné z: https://mavlink.io/en/

[7] ROS with MAVROS Installation Guide | PX4 User Guide. Redirecting to latest version of document (master) [online]. [cit. 29.12.2021] Dostupné z: https://docs.px4.io/master/en/ros/mavros_installation.html

[8] About DroneKit. [online]. [cit. 06.11.2021]. Dostupné z: https://dronekit-python.readthedocs.io/en/latest/about/overview.html

[9] Princip laserových snímačů vzdálenosti s triangulačním principem měření | Automatizace.HW.cz. Automatizace.HW.cz | Elektronika v automatizaci [online]. [cit. 29.12.2021] Dostupné z: https://automatizace.hw.cz/mereni-a-regulace/princip-funkce-laserovych-snimacu-vzdalenosti-s-triangulacnim-principem-mereni.html

[10] Čím se LiDAR liší od radaru a jaká je jeho role v autonomních vozidlech – VTM.cz. VTM.cz – Věda, technika, zajímavosti, budoucnost [online]. Copyright © 2021 Copyright CZECH NEWS CENTER a.s. a dodavatelé obsahu. [cit. 29.12.2021]. Dostupné z: https://vtm.zive.cz/clanky/cim-se-LiDAR-lisi-od-radaru-a-jaka-je-jeho-role-v-autonomnich-vozidlech/sc-870-a-195431/default.aspx

[11] RPLIDAR A3 Low Cost 360 Degree Laser Range Scanner Introduction and Datasheet [online]. Copyright © 2021 by Génération Robots. All Right Reserved. [cit. 29.12.2021]. Dostupné z: https://www.generationrobots.com/media/LD310_SLAMTEC_rplidar_datasheet_A3M1_v1.0_en.pdf

[12] Computer science | University of Toronto [online výukový materiál]. [cit. 29.12.2021] Dostupné z: http://www.cs.toronto.edu/~fidler/slides/2015/CSC420/lecture12_hres.pdf
[13] Portál pro uživatele termokamery | Průvodcem světem termokamer [online]. Copyright © Workswell s.r.o. [cit. 29.12.2021]. Dostupné z: https://www.termokamera.cz
[14] HABART, L. Využití moderních kamerových systémů při analýze silničních nehod. Brno: Vysoké učení technické v Brně, Ústav soudního inženýrství, 2013. 85 s. Vedoucí diplomové práce Ing. Vladimír Panáček. [cit. 29.12.2021].
[15] Stereo Depth Solutions from Intel RealSense. Intel® RealSense™ Computer Vision - Depth and Tracking cameras [online]. Copyright © Intel Corporation [cit. 29.12.2021]. Dostupné z: https://www.intelrealsense.com/stereo-depth
[16] Smart Powerful LiDAR Solutions | Velodyne LiDAR [online]. Copyright © Velodyne LiDAR, Inc. 2021 All Rights Reserved [cit. 29.12.2021]. Dostupné z: https://velodyneLiDAR.com
[17] Premier Choice Group – Premier Choice Group helps businesses communicate more effectively, increases business efficiency and creates an infrastructure to support growth. Founded in 2000, we have earned a reputation for our professional approach, depth of expertise, outstanding customer service, bespoke customer reports and dedicated account management. We have been awarded the ISO 9001 accreditation for our quality management systems and … [online]. Copyright © [cit. 29.12.2021]. Dostupné z: https://www.premierchoicegroup.com/wp-content/uploads/Thermal-Camera-Image.jpg
[18] Autonomous Landing of UAV Quadrotor using Reinforcement learning. Drony [online]. [cit. 29.12.2021]. Dostupné z: http://drones4geeks.blogspot.com/2017/09/autonomous-landing-of-uav-quadrotor.html
[19] Documentation - ROS Wiki. Documentation - ROS Wiki [online]. Dostupné z: http://wiki.ros.org
[20] ROS - Robotika. [online]. Dostupné z: https://sites.google.com/site/vutrobotika/navody/ros
[21] Robotics and Computer Vision Research @ TCS. [online]. Dostupné z: https://sites.google.com/site/swagatkumar/home/robotics-and-computer-vision-research-tcs
[22] Copter Home — Copter documentation. ArduPilot - Versatile, Trusted, Open [online]. Copyright © Copyright 2021, ArduPilot Dev Team. [cit. 30.12.2021]. Dostupné z: https://ardupilot.org/copter/index.html
[23] Speech Assisted Interface for Quadcopter Flight Control [online]. Copyright © [cit. 30.12.2021]. Dostupné z: https://etd.ohiolink.edu/apexprod/rws_etd/send_file/send?accession=toledo1526247041269609&disposition=inline

[24] Intel RealSense D435 review – Maurice's Musings. Access forbidden! [online]. Dostupné z: https://www.calvert.ch/maurice/2018/06/12/intel-realsense-d435-review
[25] NVIDIA Jetson Nano Developer Kit | NVIDIA Developer. NVIDIA Developer [online]. Dostupné z: https://developer.nvidia.com/embedded/jetson-nano-developer-kit
[26] Redirecting to latest version of document (master) [online]. Copyright © [cit. 31.12.2021]. Dostupné z: https://docs.px4.io/v1.9.0/images/qgc/setup/sensor_compass_calibrate_px4.jpg
[27] Pixhawk 4 | PX4 User Guide. Redirecting to latest version of document (master) [online]. Dostupné z: https://docs.px4.io/master/en/flight_controller/pixhawk4.html
[28] Setting up a Simulated Vehicle (SITL). [online]. Copyright © Copyright 2015 [cit. 31.12.2021]. Dostupné z: https://dronekit-python.readthedocs.io/en/latest/develop/sitl_setup.html
[29] Open Source Autopilot for Drones - PX4 Autopilot. Open Source Autopilot for Drones - PX4 Autopilot [online]. Dostupné z: https://px4.io
[30] Hardware-software architecture, code generation and control for multirotor UAVs. Home page [online]. Copyright © [cit. 31.12.2021]. Dostupné z: https://www.politesi.polimi.it/handle/10589/144739
[31] Getting Started With Jetson Nano Developer Kit | NVIDIA Developer. NVIDIA Developer [online]. Dostupné z: https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit#write

SEZNAM SYMBOLŮ, ZKRATEK

Zkratky:

LiDAR	Light Detection and Ranging
UAS	Unmanned Aircraft Systems
GPS	Global Positioning System
XML	Extensible Markup Language
ROS	Robot Operating System
IP	Internet Protocol
GPL	General Public License
MTOM	Maximální vzletová hmotnost
BSD	Berkeley Software Distribution
MCU	Microcontroller Unit
MAVLink	Micro Air Vehicle Link
ID	Identifikace ve výpočetní technice
API	Application Programming Interface
SITL	Software In The Loop
SDK	Software Development Kit
RGB	Red, Green, Blue
CMOS	Complementary Metal Oxide Semiconductor
SLAM	Simultaneous Localization and Mapping
TOF	Time Of Flight
CPU	Central Processing Unit
GPU	Graphics Processing Unit
USB	Universal Serial Bus
SoC	System on a chip
RAM	Random Access Memory
I2C	Inter Integrated Circuit
SPI	Serial Peripheral Interface
CAN	Controller Area Network
BLDC	Brushless Direct Current
PWM	Pulse Width Modulation

Symboly:

d	vzdálenost	(m)
c	rychlost světla	(m/s)
f	frekvence	(Hz)
$\Delta\phi$	fázový posun	(rad)
t	čas	(s)