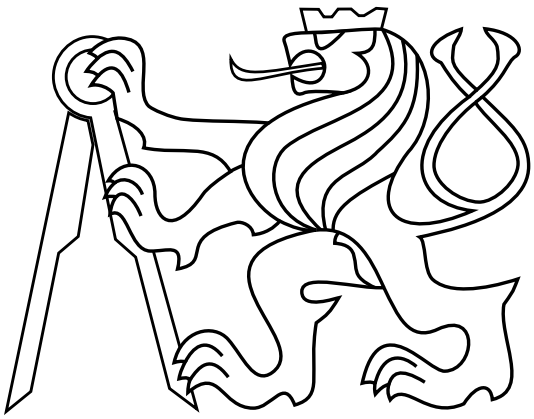
Master's thesis

Autonomous Localization of a Radiation Source With a Group of UAV in a 3D Environment

Petr Štibinger



May 2019

Thesis supervisor: **Ing. Tomáš Báča**

Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Measurement

Declaration

I declare that presented work was developed independently, and that I have listed all sources of information used within, in accordance with the Methodical instructions for observing ethical principles in preparation of university theses.

Prague, date ....

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Štibinger Petr** Personal ID number: **434636**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Measurement**

Study program: **Cybernetics and Robotics**

Branch of study: **Cybernetics and Robotics**

II. Master's thesis details

Master's thesis title in English:

Autonomous Localization of a Radiation Source With a Group of UAV in a 3D Environment

Master's thesis title in Czech:

Autonomní lokalizace zdroje záření v 3D prostředí pomocí skupiny UAV

Guidelines:

The thesis aims to explore autonomous localization of a source of ionizing radiation by a group of Unmanned Aerial Vehicles (UAVs). The Unmanned Aerial Vehicles will be equipped with the Timepix hybrid pixel detector [1], which is sensitive to ionizing radiation. The work will follow up on the previous work of the student, where the radiation source was localized by a group of 3 UAVs in a free and open environment. To improve upon the previous work, the radiation source will be localized in a complex (but already known) 3D environment, e.g., in a multi-level building. The radiation mapping will take the structure of the environment into the account. Namely, it should utilize a radiation attenuation model of the environment. The work will tackle the following points:

- Improve the current Timepix model for the Gazebo simulator: implement physics-based photon attenuation for commonly used sensor materials (Si, CdTe).
- Prepare the simulated 3D environment for the radiation localization and mapping experiments.
- Design an algorithm, which will estimate the position of the radiation source based on the measurements of the UAVs.
- Implement a system that will control the UAVs to localize the radiation source.
- Conduct simulations and conclude, whether the UAVs can serve such purpose with the Timepix sensor and if so, what are the work pre-conditions of the system.

Bibliography / sources:

[1] Llopart, Xavier, et al.: "Timepix, a 65k programmable pixel readout chip for arrival time, energy and/or photon counting measurements." Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 581.1-2 (2007): 485-494.
[2] Eric A. Wan, and R. Van Der Merwe: "The unscented Kalman filter for nonlinear estimation." Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000.

Name and workplace of master's thesis supervisor:

Ing. Tomáš Báča, Multi-robot Systems, FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **30.01.2019** Deadline for master's thesis submission: **24.05.2019**

Assignment valid until:
by the end of summer semester 2019/2020

Ing. Tomáš Báča Supervisor's signature	Head of department's signature	prof. Ing. Pavel Ripka, CSc. Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

. _____
Date of assignment receipt

_____
Student's signature

Acknowledgement

I would like to thank my supervisor, Ing. Tomáš Báča, for his great guidance, support and for sharing a lot of valuable knowledge with me. Furthermore, I would like to thank Dr. Martin Saska and the members of Multi-Robot Systems group, for their patience and help in overcoming minor obstacles, and generally creating a very positive and inspiring work environment. Finally, my gratitude goes to my family, my girlfriend and my friends for their encouragement and support not only during my studies, but throughout my life in general.

Abstrakt

Práce je zaměřena na využití skupiny bezpilotních helikoptér v úloze hledání pozice neznámého zdroje ionizujícího záření. Tato úloha je řešena v prostředí s překážkami. Podstatnou část práce tvoří vývoj nových součástí pro robotický simulátor Gazebo, díky kterým je možné vytvářet realistické simulace obsahující radioaktivní zářiče a překážky s různými vlastnostmi. K detekci radioaktivního záření slouží moderní senzor Timepix, jehož rozměry a nízká hmotnost umožňují jeho umístění přímo na palubu malých bezpilotních helikoptér. Stávající model tohoto detektoru pro simulátor Gazebo byl kompletně přepracován. Veškeré interakce záření s prostředím jsou nyní řešeny na úrovni jednotlivých částic. V prostředí simulátoru byly navrženy a otestovány algoritmy pro řízení libovolně velké skupiny helikoptér, která pomocí palubních senzorů Timepix aktivně vyhledává pozici zářiče. Teoretické předpoklady a simulace jsou podpořeny také reálným experimentem, ve kterém je pomocí helikoptéry mapována intenzita gamma záření, produkovaného izotopem Americia ^{241}Am.

Klíčová slova

Radioaktivita, ionizující záření, drony, bezpilotní helikoptéry, částicový detektor, Timepix, simulace, Gazebo, Kalmanův filtr

Abstract

This thesis tackles the task of using a group of unmanned aerial vehicles (UAVs) to locate an unknown source of ionizing radiation. The task is solved for an environment containing obstacles. A significant contribution of this work is the development of plugins for a robotic simulator Gazebo, which allow for creation of highly realistic simulations with radiation sources and obstacles of various properties. The detection of radiation is performed by a modern detector Timepix. Small size and low weight of the detector allow for it to be directly mounted onto a UAV. The current Gazebo model of this detector was completely reworked. Now the simulation takes into account interactions of individual radiation particles with the environment. The simulated environment was used to design and test algorithms for control of a group of UAVs. These UAVs utilize the onboard sensors to estimate the position of the radiation source in real time. The theoretical assumptions and simulation results are supported by a real experiment, in which one UAV was mapping the intensity of gamma radiation produced by an isotope of Americium ^{241}Am.

Keywords

Radioactivity, ionizing radiation, drones, unmanned aerial vehicles, particle detector, Timepix, simulation, Gazebo, Kalman filter

Contents

1 Introduction

The field of mobile robotics has undergone a lot of development over the past ten years. The latest advances can be attributed to three key technologies. Firstly, these are Lithium polymer batteries with better power density and lower mass than their predecessors. Secondly, brushless DC motors, which offer greater efficiency than their brushed counterpart became more affordable due to large scale production. And finally, advances in transistor miniaturization translate into construction of smaller and more powerful processing units.

All these factors have one thing in common. They allow for overall reduction in weight of the robot. With the mass reduced, mobile robots could take a new direction. Upwards, into the air. This has brought along interesting challenges and opened new possibilities for application.

Unmanned aerial vehicles (UAVs) or *drones* witnessed huge increase in popularity. Many technologies have found their way from research laboratories into our everyday life in just a few years. Nowadays, we can purchase a drone (such as the one shown in Figure 1) with a high-resolution camera, at almost any electronics store, and some of them are even equipped with "smart" functions like return to starting position, target tracking or obstacle detection.

Figure 1 DJI Mavic Pro[1]. One of the commercially available UAVs equipped with a stabilized camera and an advertised airtime of up to 30 minutes.

The technological advances apply not only in the field of mobile robotics, but in all branches of science and technology. Ever since the Second World War, humans have been making great advances in nuclear sciences. This has allowed us to harness the power of nuclear fission to generate electricity, and in the future, we may even be able to sustain nuclear fusion – the process, which powers our Sun and all other stars. However, the current state of technology has its downsides.

Aside from the most obvious dangers, posed by weaponization and nuclear power plant accidents, ionizing radiation is able to endanger human life in a completely inconspicuous way. Radiation emanating from natural sources (Uranium ore, Radon gas), spent nuclear fuel (nuclear waste) or discarded specialized equipment (smoke detectors,

[1]Source: `https://store.dji.com/product/mavic-pro-platinum`

medical X-rays) can be a stealthy killer. This is due to the fact, that our bodies are incapable of sensing the presence of radiation on their own.

Luckily, many sensors capable of detecting ionizing radiation have been developed. To accurately assess the levels of radiation, however, the sensor usually needs to be deployed close to the source. Here, mobile robots represent an ideal platform, to carry the equipment without exposing human workers to a potential danger.

This thesis aims to utilize an aerial robotic platform in a task of localizing ionizing radiation sources in a 3D environment. The main advantage, which UAVs hold over ground-based robots, is the ability to move quickly over large distances, regardless of terrain conditions. Using the UAVs in a group allows for a larger number of sensors being active at the same time, and also makes the whole system more resilient to failures through redundancy. In the near future, such system might help with assessment of health hazards in areas surrounding damaged nuclear power plants, uranium ore mines, or nuclear waste storage sites. It could also be used as a fast response to a potential public threat posed by an unauthorized stockpile of radioactive material or a stolen radiography equipment, which also tend to contain strong sources of radiation.

1.1 State of the Art

Unmanned aerial vehicles of various shapes and sizes have been used as a platform to carry scientific instruments for almost two decades now. One of the earliest uses of UAVs to carry out scientific experiments was by the US Department of Energy and the Department of Defense, using a remotely piloted fixed-wing aircraft to study Earth's atmosphere, with results published in [1]. In [2] a prototype radiation detector designed for use by a UAV is presented. A followup research, which included flight experiments with a manually controlled UAV is presented in [3], and in [4] with a UAV following a predefined trajectory.

In 2011, an earthquake and a followup tsunami had severely damaged the Fukushima Daiichi Nuclear Power plant, which lead to release of radioactive material to the surrounding area. UAVs have been deployed to monitor radiation levels in the area in an effort to minimize exposure of the cleanup workers, and to assess effectiveness of the decontamination work as well as habitability of the nearby residences [5, 6, 7, 8, 9]. The UAVs deployed to survey the area included large fixed-wing aircraft as well as smaller multirotor helicopters. In all applications, the UAVs were following a predefined trajectory.

A system for contouring an irradiated area, which utilizes multiple UAVs is presented in [10]. The UAVs move along a series of predefined waypoints or an elliptical trajectory, and the control and data processing is handled by a ground station.

Active localization or tracking of an unknown source of radio signal by a group of UAVs has been presented in [11, 12, 13]. A cooperating group of UAVs, used to localize a source of chemical pollution, is shown in [14]. In [15], a single UAV is used in the task of localizing sources of ionizing radiation with real-time trajectory adjustments based on the measurements.

The main focus in robotic radiation sensing revolves mostly around use of scintillating detectors onboard UAVs [3, 4, 7, 5, 6]. These detectors offer great detection capabilities and often also the ability to observe energy spectrum of the radiation. Analysis of the spectrum can help with determining the specific isotope releasing the radiation, which is crucial when assessing the long-term effects of pollution. However, due to the construction, scintillators tend to be bulky and heavy. This severely limits the potential

to use such detector onboard a compact UAV in an indoor environment.

The term UAV can be applied to a variety of flying vehicles, ranging from large remotely operated airplanes used by armed air forces worldwide, to small helicopter-shaped toys. In the following text, the term UAV will be used in reference to a helicopter with four (or six) vertically oriented propellers, such as the DJI F450 shown in Figure 2.

Figure 2 DJI FlameWheel F450, a medium-sized quadrotor and the primary target platform for the work presented in this thesis.

1.2 Problem definition

This thesis builds upon many years of successful reseach, conducted by the Multi-Robot Systems group (MRS) at FEE CTU[2] in Prague, in the field of stabilization and control of UAVs and their formations [16, 17, 18, 19]. The task was divided into two main parts.

The first part tackles the development of a highly realistic simulation environment, which is based on real-world physics of materials and their interaction with ionizing radiation in the form of high-energy photons. To accurately model a real world scenario, three key components need to be simulated – a source of radiation which produces the photons, obstacles which interact with the photons, and a detector which can sense the photons and provide data input for the mapping and localization algorithms. All three components need to be easily customizable, allowing the user to choose a desired thickness and material of the detector, use different radioactive isotopes as the source, and place multiple obstacles of different materials into the world. The simulation relies on material properties measured by the National Institute of Standards (NIST) available online and free of charge [20]. Simulated interactions include photon attenuation by air and obstacles, and photoelectric absorption by the detector.

The second part is focused on creating a control algorithm, which drives a group of UAVs in the task of autonomous mapping and localization of radiation sources. These tasks are performed in a 3D environment with apriori known layout of obstacles. In all the tasks, a single strong source of radiation is placed in an unknown position in the scene.

[2]Faculty of Electrical Engineering, Czech Technical University

It is assumed, that all deployed UAVs are equipped with a system capable of providing ground truth. This can be provided either by a global satellite navigation system (such as GPS, GLONASS, Galileo), RTK[3] GPS [21], beacon-based indoor navigation system [22] or an onboard relative localization system based on visual markers [23], LiDAR[4] scan matching [24], or optic flow [25]. It is also assumed, that all UAVs communicate with each other via a wireless connection (such as WiFi or Bluetooth) and share information about their position in the world, and the measurements of onboard radiation sensors. Each UAV is equipped with exactly one Timepix detector, and the construction of all UAVs, including sensor layout, is identical. Effects of the obstacles on quality of communication and self-localization are not considered in this work. For the user, output of the control algorithm is either a map of measured radiation, or a series of coordinates, which correspond to the estimated positions of radiation sources.

1.3 Contributions

This work presents three new plugins for the Gazebo simulator, which are based on realistic photon physics and allow the user to create complex scenes with multiple obstacles and radiation sources. Moreover, a control algorithm for a group of UAVs was developed. This algorithm drives the motion of the helicopters based on their sensory measurements, with the aim to improve information gained from the sensor readings with each following movement. Finally, a lightweight visualization library is also provided, allowing to draw the simulated geometric primitives in the ROS visualization tool RVIZ.

[3] Real-time kinematics
[4] Light detection and ranging

2 Preliminaries

This chapter summarizes the concepts from nuclear physics, which were considered and adopted into the simulations. It also provides an overview of the available options for ionizing radiation detection, with an emphasis on the detector of choice for this task – the Timepix. Last section is dedicated to basic principles of using the Robot Operating System (ROS) and the simulator Gazebo.

The concepts are described in a brief fashion, to provide an introduction to the topic. An interested reader may find more in-depth explanation in [26], which has also served as a great source of knowledge and equations presented in this chapter.

2.1 Isotope

When talking about radioactivity, the term *isotope* is often used. It describes variants of a chemical element, which differ in number of neutrons in the atomic core (nucleus). To specify an isotope of an element, nucleon number (sum of protons and neutrons in the core) is often added to the name of an element, e.g. ^{90}Sr or Strontium-90 to specify an isotope of Strontium with 90 nucleons. An asterisk may also be added to denote an atom is in an *excited* state (^{60}Co*), meaning the electrons do not occupy the lowest possible energy levels of the electron shell. An excited atom is not stable, as all matter tends to return to the lowest energy state over time. However, some electron configurations may form *local* energy minima. An atom can remain in this state significantly longer compared to other configurations. Such atom is therefore called *metastable*, and the metastability is indicated by adding a lowercase "m", e.g. $^{137\mathrm{m}}$Ba or Barium-137m.

2.2 Radioactivity

Radioactivity is a property of unstable isotopes. An unstable atom tends to release excess energy by emitting *radiation* in the form of charged particles, electromagnetic waves or neutrons. This naturally occurring process is called radioactive (or nuclear) decay. Released radiation varies not only in composition, but also in ability to interact with other matter. Large and charged particles tend to be much more reactive. Therefore, this kind of radiation will be absorbed by the environment much sooner. On the contrary, electromagnetic waves without charge or mass tend to reach (penetrate) much much further into materials. The different types of radiation and their penetrative abilities are shown in Figure 3. Strongly penetrating radiation, such as gamma or neutron, is also referred to as "hard" radiation.

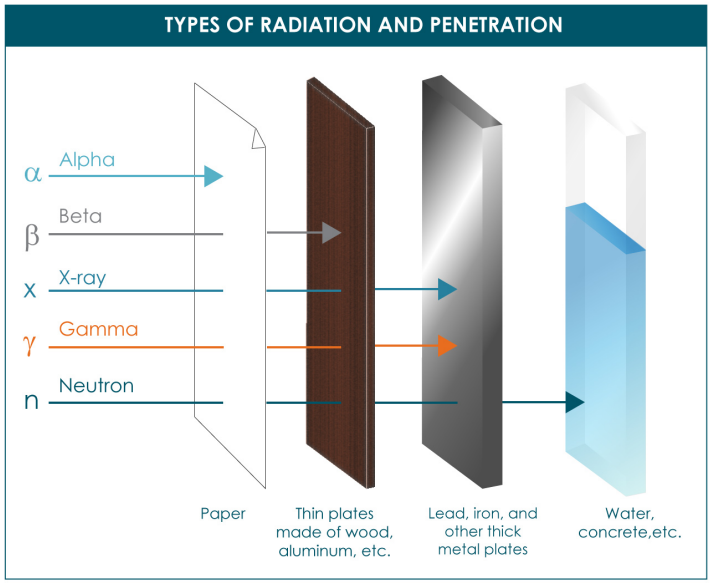


Figure 3 An overview of different types of ionizing radiation, and an efficient way to stop them. Source: `https://www.mirion.com/learning-center/radiation-safety-basics/types-of-ionizing-radiation`

Alpha radiation is the release of alpha particles, consisting of two protons and two neutrons (Helium nuclei). Alpha particles are the heaviest of all nuclear decay products, although even larger ions can occur as a result of a rare process called *spontaneous fission*. The two protons provide alpha particles with a strong positive charge, which makes them extremely reactive with other matter, while at the same time also limits their range and penetration. Human skin, or even a single piece of paper can effectively stop the radiation, which makes alpha sources the most difficult to detect from a distance. Alpha sources are generally not considered dangerous to humans, unless they are ingested.

Beta radiation is the release of electrons (or positrons) at a high velocity. The released particle is lighter than the alpha particle, and has a weaker electrical charge, which makes it less reactive. This causes beta radiation to reach further into materials. It is able to penetrate human skin, but can be effectively stopped by an Aluminum sheet or a wooden board. However, more dangerous than the released electrons themselves, are electromagnetic emissions caused by their interactions with other matter. An electron traveling at a high velocity is able to produce secondary gamma radiation by releasing another electron from an atom, or by deflections and braking induced by other atoms (bremsstrahlung). In the case of positron emission, gamma radiation is released when a positron collides with an electron (annihilation). These effects need to be taken into account when working with beta emitters or designing protective cover.

Gamma radiation is the release of high energy photons. It is a form of electromagnetic radiation, without mass or electrical charge. Gamma rays pass through skin, wood and even concrete, which makes this radiation significantly more dangerous than alpha or beta. Elements with high atomic number and large nuclei are usually used as gamma ray shielding (Lead, Uranium or Bismuth). Higher energy of the photons usually translates to harder radiation, however the exact relation depends heavily on the obstacle material.

X-rays are also a form of electromagnetic radiation, but differ from gamma rays in energy or the physical process, which produced them. Since a general rule to separate

X-rays and gamma rays does not exists, the term gamma ray will refer to any photon produced by a radioactive source in the context of this work.

Neutron radiation is the release of neutrons, and usually occurs when an atom is split by nuclear fission. The released neutron has a chance of making other atoms unstable by crashing into the core and causing them to decay as well. This property is commonly used in nuclear fission reactors, where neutrons released by splitting of an Uranium atom initiate the fission process in new atoms. Free neutrons bear no charge, therefore neutron radiation also reaches further than alpha or beta, and is also considered hard. Opposite to gamma radiation, the most efficient way to stop neutrons is with light nuclei, such as Hydrogen, which does not destabilize by the neutron impact. The most common material used to absorb neutron radiation is water.

Since the SI unit of energy, Joule, is too large to describe processes on the scale of atoms, the energy of individual particles is usually measured in electronvolts (eV). The conversion from electronvolts to Joules is defined as:

$$1\,\mathrm{eV} \approx 1.602 \times 10^{-19}\,\mathrm{J}\,. \tag{1}$$

2.3 Health risks

Ionizing radiation is harmful for humans and most of other known living organisms. It causes degradation of living tissues by killing cells. The resulting health issues vary, depending on the duration of radiation exposure, energy of the radiation and the affected body part. Some of the symptoms, like skin burns, are nearly immediate. These are usually the result of a short-term exposure to high energy radiation. Other symptoms, like headaches, nausea and fever are typically results of exposure to lower-energy radiation. These symptoms can occur several hours or even days after the actual exposure, after the body starts to dispose the damaged cells, and can still be lethal. Exposure to ionizing radiation can also cause mutations in DNA and increase the risk of cancer and heart disease.

Since human body is not able to distinguish a radioactive isotope from a stable one, some of radioactive isotopes of common elements may accumulate in the body. These isotopes then continuously irradiate the body from within. In this case, even alpha and beta sources pose a significant threat to the tissue in direct contact with the radiation.

The amount of ionizing radiation, which a body receives over time, is called radiation dose. It represents the total energy of all ionizing particles absorbed by a material per unit of mass. An SI-derived unit of radiation dose is Gray (Gy), which equals to absorbed energy of 1 Joule per kilogram. To put these numbers into perspective, an average dose absorbed by a person living in the USA, according to [27], is 5.6 mGy per year. One X-ray imaging procedure equals to a dose of 1–3 mGy. A limit considered harmless when working near radiation sources is 50 mGy per year.

2.4 Properties of radioactive materials

Nuclear decay is a *stochastic* process which ends with an atom decaying and releasing radiation. A method to predict the exact decay time for a specific atom is yet to be found. Therefore, statistics and probability is used to describe quantitative properties of radioactive materials. In the context of this work, the two most important properties are *half-life* and *activity*.

Half-life represents a time period, after which all atoms in a sample will have had exactly 50% chance of decaying. On average, half of the particles decays during this

time. It is the most common property used to determine stability of an isotope. Some isotopes produced by nuclear fission in power plants (such as Cesium-137 or Strontium-90) have a half-life of around 30 years, which makes contamination by these a long-term concern, as it takes nearly 130 years before 95% of the material naturally transforms into stable isotopes. However, this transformation is often not straightforward. An atom may undergo a sequence of decays into unstable states before finally reaching stable state. These sequences are called *decay chains*, and two specific examples are presented later in this chapter.

Activity describes the rate of radioactive decay in number of decay events per unit of time. This, however, does not directly translate to number of emitted particles, as many isotopes have more than one possible outcome of the decay process. When describing a specific sample with a given amount of the material, *specific activity* can be calculated from a known half-life of an isotope and the mass of the sample. The unit of activity is Becquerel (Bq). One Becquerel equals to one decay event per second. The specific activity A is calculated as:

$$A = \frac{m}{m_a} N_A \frac{ln(2)}{t_{1/2}} \ , \tag{2}$$

where m is mass of the sample in grams, m_a is mass of one atom of the isotope in grams, $N_A \approx 6.022 \times 10^{23}$ is the Avogadro's constant and $t_{1/2}$ is half-life of the isotope in seconds.

To provide the reader with a bit of perspective, one gram of pure Cesium-137 has a specific activity of approximately 3.215 TBq (3.215×10^{12} Bq).

This thesis is focused mainly on two common radioactive isotopes, Cesium-137 and Americium-241, which are simulated by the plugins presented in Chapter 3. These isotopes were chosen because of their abundance in the real world (relative to other radioisotopes), and severe health risks posed by the emitted radiation.

2.4.1 Cesium-137

Cesium-137 is a radioactive isotope of Cesium. It is commonly produced as a result of Uranium nuclear fission, which is used as a source of power in many nuclear power plants and nuclear weapons.

As an element, Cesium is located in the first column of periodic table, in a group called *alkali metals*. All alkali metals are extremely reactive, which is a consequence of having only one electron in their valence[1] shell. Pure Cesium dissolves in water to form Cesium Hydroxide in a violent reaction, which generates extreme amounts of heat. After dissolving, it easily contaminates soil and sources of groundwater.

The radioactive isotope ^{137}Cs has a half-life of approximately 30.17 years, which makes it one of the longest living components of radioactive waste. The decay products of ^{137}Cs are shown in Figure 4. Although the Cesium itself releases only beta radiation, nearly 95% of all atoms end up transforming into a metastable isotope of Barium, ^{137m}Ba. This isotope is relatively short-lived, with a half-life of only 2.5 minutes, after which the atom releases a 662 keV gamma ray and decays into the final state – stable isotope ^{137}Ba. The half-life of ^{137m}Ba is significantly shorter than half-life of ^{137}Cs, and the decay is often considered as instantaneous. For this reason, ^{137}Cs is usually considered a source of gamma radiation, even though it technically does not release any gamma rays directly.

[1] Furthest from the core

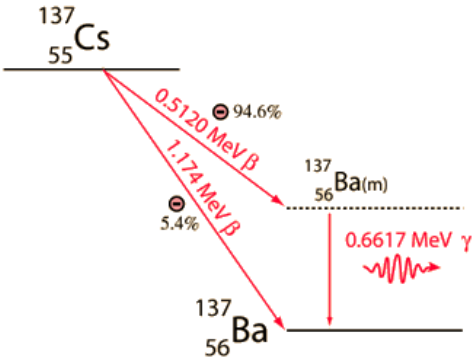


Figure 4 Decay chain of Cesium-137. The isotope directly emits beta radiation in the form of high energy electrons. However, a majority of these atoms transforms into a metastable isotope Barium-137m with a half-life of 2.5 minutes, which decays into stable Barium by releasing a 662 keV gamma ray. The half-life of ^{137m}Ba is so short, compared to the 30 years of ^{137}Cs, that the Cesium isotope is commonly considered a gamma ray source. Source: `http://hyperphysics.phy-astr.gsu.edu/hbase/NucEne/imgnuk/cs137decay.gif`

2.4.2 Americium-241

Americium is a heavy[2] radioactive element with no stable isotope. It occurs as one of the products of nuclear fission, and also as a product of naturally decaying Plutonium. The isotope ^{241}Am has a half-life of 432.2 years, and is a source of alpha particles as well as weak gamma rays with energy of 52.9 keV.

This isotope is commonly used in ionization chambers of household smoke detectors. The amount of Americium used in these detectors is relatively small and is considered safe. However, disposing a large number of these detectors in a landfill can still cause environmental damage, which will be persistent for many centuries due to the longevity of the isotope. ^{241}Am is also one of the longest-lived components of nuclear waste, and its presence will be still traceable around nuclear test sites and damaged power plants, after the other byproducts have already decayed.

The decay process of ^{241}Am is illustrated in Figure 5. The isotope has two possible modes of decay. Both of them release a high energy alpha particle accompanied by a low energy gamma ray. The resulting product is always Neptunium-237, which is also radioactive, and the following decay chain to a stable isotope of Lead is very long with many transient states. However, ^{237}Np has an exceptionally high half-life of 2.14 million years. Therefore, the other products can be omitted, as the probability of a next decay in the chain decreases dramatically.

[2] Having a large number of protons, but also high density

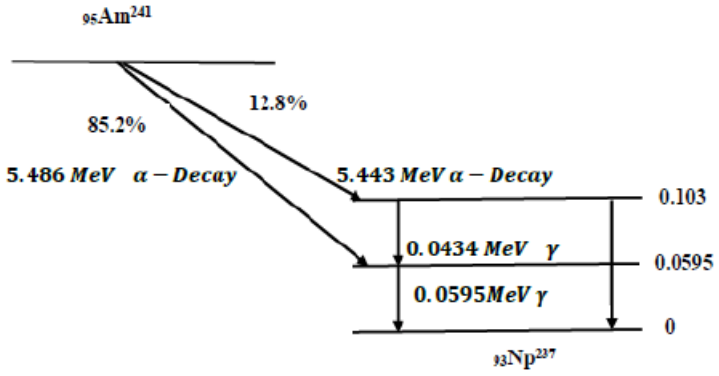


Figure 5 A reduced decay chain of Americium-241. The isotope has two possible modes of high energy alpha emission, which are both accompanied by a weaker gamma emission. The presence of this isotope is usually determined from a prominent peak in a radiation energy spectrum located at 59.5 keV. The highly energetic alpha particles are easily stopped by one sheet of paper, or by a few centimeters of air, which makes them undetectable at a larger distance. Source: `http://www.rroij.com/articles-images/pharmaceutical-analysis-decay-5-4-22-g001.png`

2.5 Detecting radioactivity

Human body lacks the ability to sense ionizing radiation on its own, therefore use of specialized detectors is required. The available sensors vary in construction as well as the physical principles used in the detection process.

2.5.1 Ionization chambers

Ionization chambers are among the oldest radiation detectors, and their construction is one of the simplest. The main part of the detector is a gas-filled tube and an electric field introduced by an external voltage source. Radiation passing through the tube ionizes the gas, which leads to generation of charged ions in the chamber. These ions cause disruptions in the electric field. By measuring these disruptions, presence and intensity of the radiation can be determined. The widely used Geiger counter is an example of a detector with an ionization chamber. These detectors are not very suitable for miniaturization, because of low density of atoms in gases, which results in a lack of detection medium at small sizes.

2.5.2 Scintillation detectors

Scintillation is an emission of visible light caused by ionizing radiation passing through certain materials. The wavelength of emitted light varies, based on the excitation energy. For this reason, scintillating detectors are often used not only for detection, but also for spectroscopy of the radiation. Studying the spectrum can reveal more information about the source. The complex chain of events during the detection process (radiation is transformed into visible light which is transformed into electrical signal) naturally introduces energy losses. Therefore the resolution of a final spectrum cannot reach the levels of more straightforward detectors.

Scintillators are also heavier than other types of detectors, as they require a solid piece of scintillating material with optical amplifiers and sensors surrounding it. The

scintillating material may be either solid, liquid or gaseous. However, solid-state detectors have the most densely packed atomic structure, which provides the highest chance of interaction with incoming radiation.

2.5.3 Semiconductor detectors

Semiconductors rely on presence of charge carriers in the form of electrons and holes in the atomic lattice. Similar to the ionization chamber, radiation passing through a semiconductor creates new electron-hole pairs in the material. This event can be directly converted into an electrical signal by a semiconductor diode.

The main advantage of semiconductors is a low ionization energy, which means even a low-energy radiation can produce a lot of charge carriers. This offers great energy resolution, as well as the ability to detect single particles.

Timepix

The Timepix belongs to a group of semiconductor detectors called *pixel detectors*. It comprises of a matrix of 256×256 pixels, which are tiled on a single block of detection material (Si, GaAs or CdTe) of an area 14.08×14.08 mm. Every pixel works as an individual detector, and the individual pixels are bump-bonded to the readout ASIC[3]. The readout can be performed in either serial mode (full image under 10 ms) or in parallel mode (full image under 300 µs). The newest version of the detector, Timepix3, is also able to perform readout of individual pixels independently as soon as an event is recorded by the pixel [28]. This property can be very useful in the field of mobile robotics, as the received radiation data can be passed to a control algorithm immediately, allowing for a nearly real-time reaction.

The Timepix chip has been developed by the Medipix collaboration at CERN[4] as a new, enhanced version of detector Medipix. The Institute of Experimental and Applied Physics of CTU in Prague has developed the USB readout interface for the detector [29, 30], which created new application options for the Timepix. Notable examples include a Timepix-based miniaturized X-ray telescope onboard the nanosatellite VZLUSAT-1 [31, 32], imaging of high-energy experiments at the LHC[5] [33], monitoring of radiation levels onboad the ISS[6] [34] or non-destructive painted art analysis [35].

The detector is able to operate in one of three modes:

- Medipix mode – counts the number of events within a predefined energy range over the duration of one exposure
- Time of Arrival (ToA, Timepix mode) – stores the exact time of the first event exceeding an energy threshold
- Time over Threshold (ToT) – integrates the energy over threshold deposited in a pixel over the duration of one exposure

Exposure times can range from a few microseconds to several hours, provided the detector is properly calibrated and the readout interface offers sufficient data bandwidth. Small size and close proximity of the pixels, combined with the low excitation energy of semiconducting materials, can cause multiple pixels to be affected by a single

[3] Application-specific integrated circuit
[4] The European Organization for Nuclear Research
[5] Large Hadron Collider
[6] International Space Station

particle. This is shown in Figure 6, where multiple tracks left by different particles are visible. The size of tracks depends on the bias voltage of the detector, with higher voltage resulting in narrower tracks.

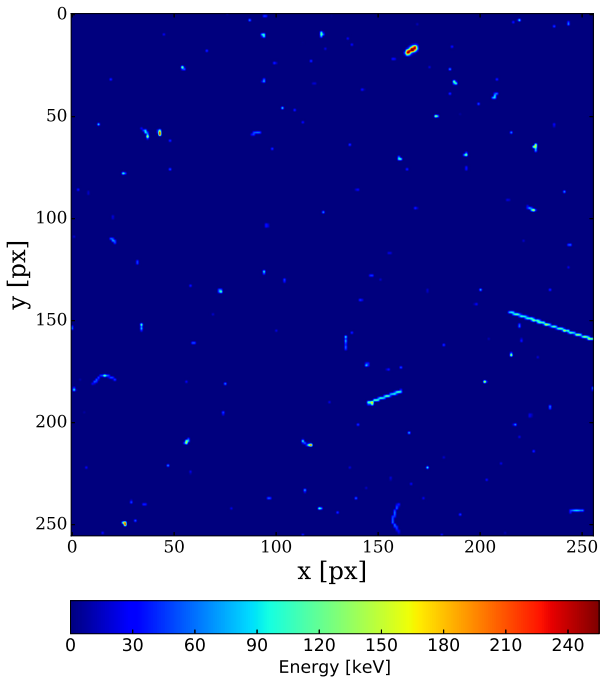
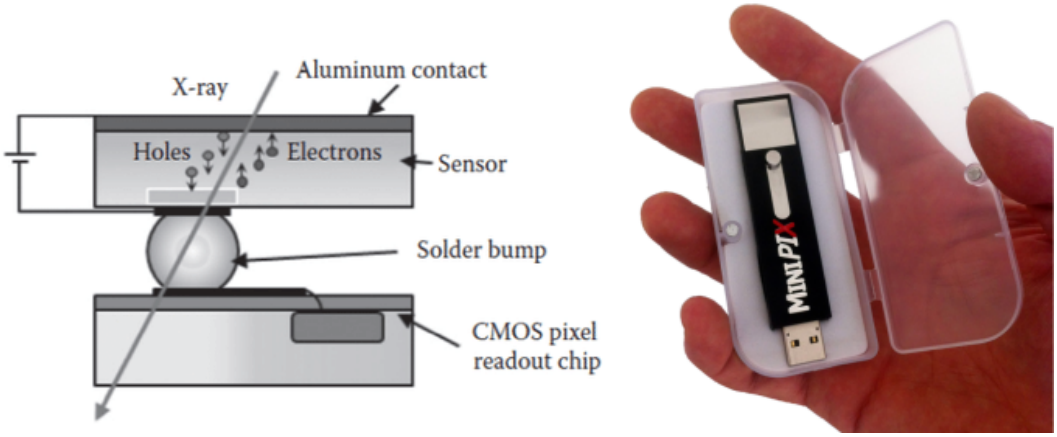


Figure 6 A full image taken by Timepix onboard the VZLUSAT-1 in Low Earth Orbit. The image features numerous tracks made by different particles.

Analyzing the tracks with image recognition software can provide more information about the source of the radiation. Longer tracks or blobs of high energy are caused by heavy ions or electrons, whereas gamma and X-rays leave tracks at only one or two pixels.



a) Inner structure of one pixel **b)** MiniPIX – USB radiation camera

Figure 7 Timepix detector – the inner structure of a single pixel is shown in **a)**. A typical Timepix chip consists of 65536 pixels. A miniaturized USB interface equipped with the Timepix detector is shown in **b)**. The metallic surface of a Timepix chip can be seen in the upper part of the device. Image sources: [36], `http://advacam.com/776.html`.

Using the detector onboard a small UAV is motivated by the size of the device. The smallest available configuration, called MiniPIX, is shown in Figure 7b. MiniPIX is comparable to a flash drive both in size and weight, and is capable of taking up to 30 images per second. The device uses USB2.0 for data transfer and power supply.

The detector can be easily used with any robotic platform using the Robot Operating System (ROS) via a driver called Rospix [37]. This software allows the user to operate multiple Timepix or Medipix devices with a single instance of Rospix running. It provides the user with the options to dynamically change parameters of the sensor (such as exposure time, energy threshold or acquisition mode) by sending commands through ROS communication infrastructure. Moreover, Rospix also provides some resilience to communication loss, as it automatically tries to reestablish contact with the detector, and upload the last used configuration to the control board. These features are extremely valuable, especially when the detector is mounted on a UAV moving in an irradiated area, far beyond the reach of a human operator.

The Rospix package also includes a series of additional tools, most notably a Random forest-based image classifier [38]. This classifier is able to distinguish individual particles based on tracks left in the image of the detector. In the case of this work, both ^{137}Cs and ^{241}Am emit predominantly gamma radiation. The image classifier can be used to filter out all other tracks besides the gamma photons, to diminish the interference of a natural radiation background.

2.6 Robot Operating System

The Robot Operating System (ROS) is a free open source framework designed for use in robotic systems. These systems are often composed of many different components developed by various manufacturers. To allow interaction between the components (sensors, actuators, displays) and a control software, *middleware* such as ROS is very useful. It provides a communication infrastructure as well as hardware abstraction for the entire system. This way, the user is able to interact with the hardware by using *services* and sending or receiving *messages*.

Despite the naming, ROS is not an operating system on its own. It is recommended to use operating system Ubuntu to run ROS. Other operating systems, like Windows or Mac OS may be used as well with varying levels of support [39].

Supported programming languages in ROS are C++, Python and Lisp. However, there are also experimental libraries which allow the use of other programming languages (Java, Lua). Matlab can also be connected to the ROS infrastructure with the use of Robotics System Toolbox [40].

2.6.1 Infrastructure

ROS implements a *master-slave* system. The master is represented by a ROS server and most often is running on the main computer. All other components in the system work as slave devices. The main job of a master is to register new *nodes*. A node can be any active ROS executable, either a computer program or a device driver.

The nodes communicate with each other by sending *messages*. A node can *publish* messages to a *topic* with an arbitrary name. Other nodes can *subscribe* the topic to receive all messages published to this topic. One node can publish to multiple topics and at the same time, multiple nodes can listen to the same topic. It is worth noting, that the messages do not need to pass through the master, if a direct link between the two communicating devices exists.

Topics do not provide confirmation of message delivery, and are therefore more suited for transporting a data stream, such as a sensor output. The communication uses either a TCP[7] or UDP[8]. The protocol is chosen automatically based on type of the connection between the two nodes. Besides messages, ROS nodes can also communicate via *services*. A service is a confirmed type of communication, and is therefore more suitable for issuing critical commands, such as providing a motion setpoint or terminating operation. Services always use TCP protocol.

A protocol called *multimaster* is required in order to use multiple independent robots, such as in a team of UAVs. In this scenario, each onboard computer acts as a master to the UAV's sensors, and only selected topics are shared with the other masters. This not only saves energy, but also prevents overloading of the communication channel.

2.6.2 Rosbag

Rosbag is a data format, which allows the user to *record* all messages sent over chosen ROS topics. It is a common tool used to collect data from a robotic system for further analysis, and easy reproduction of certain scenarios. A recorded rosbag can later be *played*. Playing a rosbag will reproduce the stored messages into their corresponding topics with the exact same time intervals as during the recording. The speed of the playback can also be increased or decreased, and even single steps can be performed. Using the API[9], a recorded rosbag may also be loaded as a structure into a C++ or Python program for further processing or editing.

[7] Transmission Control Protocol – requires confirmation → higher reliability, higher latency
[8] User Datagram Protocol – no confirmation → lower latency, unreliable
[9] Application programming interface

3 Gazebo

Gazebo is a free, advanced robotics simulator. It is fully compatible with ROS and offers a Graphical User Interface (GUI), rendering of the scene using OGRE3D[1], real-time physics (ODE, Bullet, Simbody or Dart), visualization of common sensors (contact sensors, monocular cameras, depth cameras, laser rangefinders) and use of custom 3D models [41]. Both ROS and Gazebo are being constantly developed by the Open Source Robotics Foundation[2]. A screenshot of a simulation running in Gazebo is shown in Figure 8.

Users can also extend the simulator with custom plugins written in C++. The plugins can be attached to a specific object, or to the world to provide global effects. This approach is similar to using scripts in game engines such as Unity[3] or Unreal Engine[4]. Inside the plugin, a ROS node can be created to send messages through the ROS infrastructure.

The plugins are also able to communicate via Gazebo messages defined by Google Protobuf [42]. The main advantage of using Gazebo messages over ROS messages is a much lower latency. Most of the plugins run on the computer where Gazebo server is launched, and the messages therefore do not need any transport layer, since most of the variables are stored in the same system memory. ROS, on the other hand, preserves the communication over TCP or UDP for all nodes, which can lead to delays.

Figure 8 A screenshot from Gazebo showing the default graphical user interface (GUI) with a two F550 helicopters and a few obstacles with various textures. The UAVs bear a bright green marker. This indicates, that a UAV is carrying a Timepix sensor.

[1] Open Source Graphics Engine
[2] `https://www.openrobotics.org/`
[3] `https://unity.com/`
[4] `https://www.unrealengine.com/`

3.1 Simulated environment

This chapter explains in detail the implementation process of all Gazebo plugins, which were created for the purposes of this work. Gazebo does not provide any plugins allowing for simulation of ionizing radiation, and the only comparable research presented in [43] includes a Gazebo simulation of radiation to only a very limited range of 1.5 m. Therefore, all herein presented plugins are provided to the community as open source via a GitHub repository[5]. For this reason, this section also focuses on how to use these plugins and how to adjust the parameters to create different scenarios.

3.2 Gazebo plugins

The creation of custom plugins is encouraged by Gazebo developers and the official website even provides basic tutorials and examples. The simulator provides a basic code skeleton for implementation of 6 different classes of plugins, from which the custom plugins inherit. These classes are:

- World,
- Model,
- Sensor,
- System,
- Visual,
- GUI.

World, model and sensor plugins are bound to a specific object of the scene and add some functionality beyond the default provided by Gazebo. For example, world plugin can be used to add wind through the physics engine, model plugin can add animation to a model and sensor plugin can enhance a basic camera with automatic exposure control. System plugins can be used to provide the user more control over the simulation via command line, or to redirect some topics to a file output. Visual plugins affect the rendering of a scene, and can be used to add reflective materials or sensor visualization. GUI plugins provide additional windows and panels to the existing user interface (default GUI shown in Figure 8).

It is worth pointing out, that the type of a plugin does not limit its use to a specific task. A model plugin can be still used to affect scene rendering and a sensor plugin can still come with an addition to the simulator GUI. The types of plugins just make specific tasks easier by having access to some of the environment variables (e.g., a pointer to a model or the object hierarchy tree) from the start.

In this thesis, only model plugins are used for implementation. It was decided not to use a sensor plugin for simulation of the Timepix detector, as the default sensors do not provide any similar functionality. The only sensor which is remotely close to a pixel detector would be a visible light camera, yet it would still require heavy modification, and it would end up being simpler to just build it from the ground up. This is exactly the approach taken in this work.

The skeleton of a model plugin provides several virtual methods, which can be overridden to provide desired functionality. The most important methods include:

- `void Init()` – always called as the first method of the plugin, called only once

[5] `https://github.com/rospix`

- `void Load (physics::ModelPtr _model, sdf::ElementPtr _sdf)` – called only once after a world is loaded, connects the plugin to the simulator environment. The argument `_model` points to the physical object this plugin is attached to. This provides access to properties such as position, orientation or velocity of the object. The model parameters are updated by the simulation loop with a default rate of 100 Hz. Argument `_sdf` points to the `<plugin>...</plugin>` section of the world file, which attaches this plugin to an object. This provides access to attributes, which can be changed in the sdf file without the need for compilation.
- `void Reset()` – usually triggered by user interaction with the object through Gazebo GUI, which cause the model to be temporarily unreachable by the simulation loop (actions like cut and paste).

Each plugin can also be connected to Gazebo's *event* system. An event is an action, which is automatically performed by the simulator regardless of the type of simulation. These events either occur only once per simulation (a world has been created), get triggered by user interaction (new entity is added), or happen periodically (physics update, scene rendering).

3.3 Gazebo messages

Before we dive into the details about each plugin, let us take a look at the communication infrastructure. As mentioned before, it is much faster to use internal Gazebo topics rather than ROS topics, because it dramatically reduces the transport layer for many variables, which are shared by multiple plugins.

Two topics are used to broadcast the status of the static radiation-related elements in the scene – sources and obstacles. These topics are named `/radiation/sources` and `/radiation/obstacles`, and use custom Gazebo messages.

The messages are provided in a separate package `gazebo_rad_msgs`. As mentioned previously, the structure of a Gazebo message is defined by Google's Protocol Buffer. The body of a message is declared in a `.proto` file. An example of this declaration is shown in Listing 1. The example shows a message composed of 6 variables of different data types. All of those variables are marked as required, meaning this field has to be filled in order to properly format and send the message. Each field is assigned a unique index number. This package defines Gazebo messages of types RadiationSource and RadiationObstacle.

```
1  syntax = "proto2";
2  package gazebo_rad_msgs.msgs;
3
4  message RadiationSource
5  {
6    required double x                    = 1;
7    required double y                    = 2;
8    required double z                    = 3;
9    required string material             = 4;
10   required double activity             = 5;
11   required  uint32  id                 = 6;
12 }
```

Listing 1 An example of a Protobuf message definition. This message type is broadcasted by all radiation sources (`gazebo_rad_source`) to provide the other nodes information about the position, activity and material of the source. The messages are sent to topic `/radiation/sources`.

3.4 Radiation source

This plugin, located in the package `gazebo_rad_source`, allows the user to simulate various radioactive materials. The properties of a material can be adjusted by two parameters – material name and specific activity. These parameters can be adjusted directly in the `model.sdf` file, which defines the radiation source as a Gazebo object in an xml[6] format. Changing the parameters does not require new compilation, however the simulation needs to be restarted.

When active, this plugin continuously broadcasts messages of type `RadiationSource` (full message shown in Listing 1). The message includes coordinates of the source in frame of the simulation world (global coordinates), material name which defines the isotope (e.g., "Am241"), specific activity in Becquerels and a unique ID assigned to the model by Gazebo on spawn.

The plugin is subscribed to a WorldUpdate event, which runs by default at a frequency of 100 Hz. On every world update, a new RadiationSource message is generated and published. All sources publish the messages to a topic `/radiation/sources`, and the unique ID component is used to distinguish individual sources. Publishing the message in every update step allows creation of a very complex and dynamic environment, even with moving sources of radiation.

The source appears as a cube in Gazebo GUI, however, the cube serves only as a helper object, to be clickable and easily recognized by the user. For computational simplicity, the simulated radioactivity originates from a point source. A more widespread contamination can be approximated by placing multiple weaker sources into various places in the scene.

Changing the properties of a Gazebo object unfortunately requires the entire simulation to be restarted. For user convenience, the radiation source plugin can be configured *on the fly* by using standard ROS infrastructure. The user can publish a message to topics `/radiation/debug/set_activity` and `/radiation/debug/set_material` from terminal. The message has to contain the unique ID of a source, and a new value for the changed property.

3.5 Radiation obstacle

This plugin is located in the package `gazebo_rad_obstacle` and allows the user to add obstacles, which act as regular collision objects, and also interact with the radiation. Regardless of their actual shape, the plugin considers each obstacle to be a simple cuboid. Their defining parameters are width, depth, height and material. The type of material affects the obstacle's ability to stop ionizing radiation. Some of the most common materials are directly implemented into a material library within the plugin. These include *concrete, water, glass, air* and *wood*. Other materials can be added by extending the library and compiling the package again, or by directly filling in the material properties into the `model.sdf` file which defines the obstacle as a Gazebo object.

The SDF format, which is used to define gazebo objects, supports use of inline scripts written in Embedded Ruby programming language. This allows the user to create local variables, access files, and fill in the structure of an SDF file dynamically. This is heavily utilized in the `model.sdf.erb` and `model.config.erb` files defining each radiation obstacle.

[6] eXtended Markup Language

New obstacles can be generated with ease in the folder `gazebo_models` by creating a copy of an existing obstacle and changing the name of the new folder. When a certain naming convention is kept, the embedded script is able to parse the folder name and automatically update the material and dimensions of a newly created obstacle. The convention requires the name to follow a structure: `material_depth_width_height`.

The files using Embedded Ruby require compilation before they can be inserted into the world. Since Gazebo does not compile these files automatically, an automated bash script is provided in the `gazebo_models` folder, to automatically process all model definitions within the folder.

The obstacles publish messages of type `RadiationObstacle` on every simulation update step. They all use a shared topic `/raditation/obstacles`, and the individual objects are distinguished by a unique object ID within the message. Unlike the source plugin, obstacles are actual 3D objects, therefore the message also has to contain their coordinates and orientation in the world frame.

3.6 Timepix model

This plugin is provided in the package `gazebo_timepix`. Apart from the source and obstacle plugins, which passively broadcast their status, the Timepix model handles the actual simulation of gamma radiation. As was previously mentioned in Section 2.4, even a single gram of ^{137}Cs produces on average more than 3×10^{12} particles every seconds. Simulating that many particles in real-time is beyond the abilities of conventional CPUs. However, the computational requirements can be dramatically reduced by several implementation tricks, which are described later in this section.

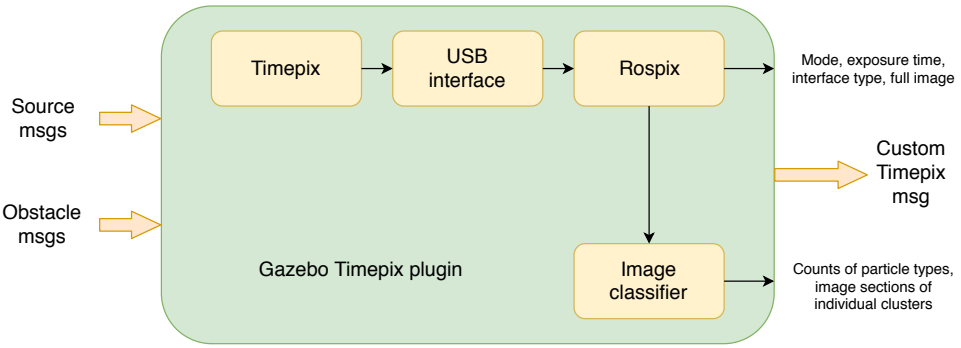


Figure 9 A diagram showing the real components, which the Gazebo Timepix plugin substitutes. The output message does not resemble the real output of the hardware. Instead, the message contains parts of both real outputs, namely the exposure time in seconds, number of captured photons, dimensions of the detector and the Gazebo time of the sensor readout. The structure of this message is defined in package `gazebo_rad_msgs`.

The output of the plugin is a custom ROS message of type `Timepix`, defined in package `gazebo_rad_msgs`. This message is simpler than the real output of the sensor, because the plugin does not simulate other particles than photons, and no full image from all pixels is created. We can think of the plugin more as a real sensor operated by the Rospix driver with the image classification running in real-time, as illustrated by Figure 9. The output message contains the following data: count of photons in the image, exposure time in seconds, dimensions of the detector in meters, simulation time at the moment of readout and a unique ID assigned to the model by Gazebo.

As mentioned previously, radioactive sources are stochastic systems, and the exact time and direction for a single particle emission cannot be determined. On a large scale, the specific activity can be considered as an invariable frequency of emissions. We assume a homogeneous material as the source, therefore the spatial distribution of emissions can be considered uniform. Under these assumptions, it is clear that a vast majority of all particles will end up completely missing the detector, as illustrated by Figure 10 for a 2D case.

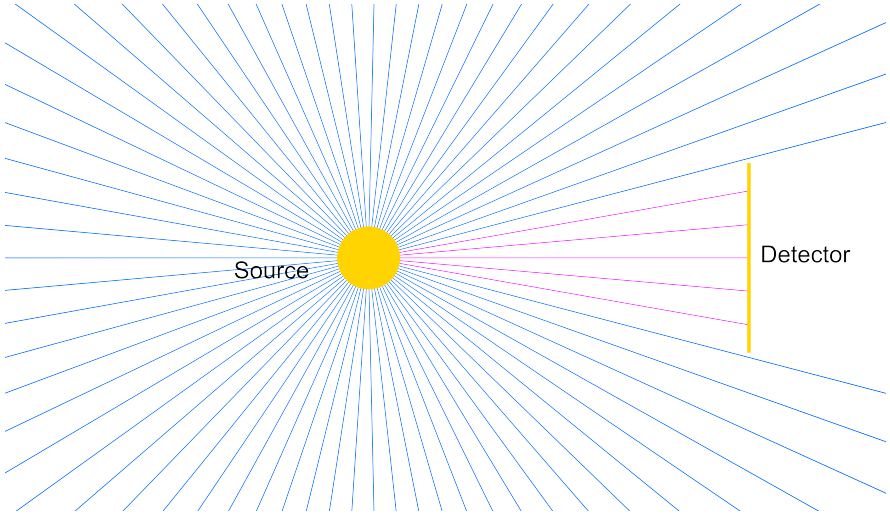


Figure 10 A 2D illustration showing that a vast majority of all gamma rays released by a radiation source miss the detector surface. The simulation complexity of can be reduced by only casting rays in the direction of the detector.

3.6.1 Ray tracing

Since the photons travel at a straight line from the source to the detector, the problem can be solved by *ray tracing* algorithms. It is a common technique used in 3D graphics and rendering. It relies heavily on projections of lines (rays) between various objects and the camera, which is where the name of the technique originates. Nowadays, it is mostly used to display 3D scenes with a high degree of realism in lighting, shadows and reflections.

It has been established earlier, that gamma radiation is a stream of photons. Just like with photons of visible light, we can use ray tracing algorithms to create realistic results with gamma rays. We can imagine every obstacle as a partially transparent object, which reduces the number of photons passing through. This "transparency" is derived from the ability of each material to absorb gamma radiation. In a similar fashion, we can imagine larger amounts of radioactive material to be stronger light sources.

The algorithm used for ray tracing in this thesis is based on finding an intersection of a line and a plane. This geometry problem has three possible outcomes – no intersection exists, exactly one intersection point exists, all points on the line are intersections. By extension, this algorithm is also used to find intersections of box-shaped objects, and calculate the length of tracks left by photons in the cuboid.

3.6.2 Implementation steps

The percentage of rays hitting the Timepix depends on the apparent size of the detector from the point of view of the source. This apparent size is affected by relative distance and orientation between the source and the detector, and by the actual size of the detector.

In this work, only the block of semiconductor material is considered as the actual detector. It is simulated as a cuboid of 14.08×14.08 mm in width and height, with a customizable thickness (set to 300 µm by default). With real detectors, the thickness of the material ranges from 100 µm to 1 mm. Experiments with a real radiation source have shown, that the thin plastic case does not introduce any measurable shielding of the radiation (see Section 5.1.2 for results of the experiment). Therefore, the outer case is not simulated, and the model is equally capable of detecting radiation hitting the front exposed face, as well as the "hidden" rear face and the sides.

Assuming the source emits radiation in all directions, the total ray output is equal to the number of rays hitting the surface of a sphere centered around the source. In a lossless environment, the number of rays per second is equal to the specific activity of the source. Surface area of the detector can be expressed as a portion of this spherical surface by using *solid angles*. In 2D, an arc is a portion of a circle defined by a given central angle. In 3D is a portion of spherical surface is defined by a corresponding central solid angle. The unit of a solid angle is *steradian*. The analogy of an angle and a solid angle is illustrated by Figure 11.

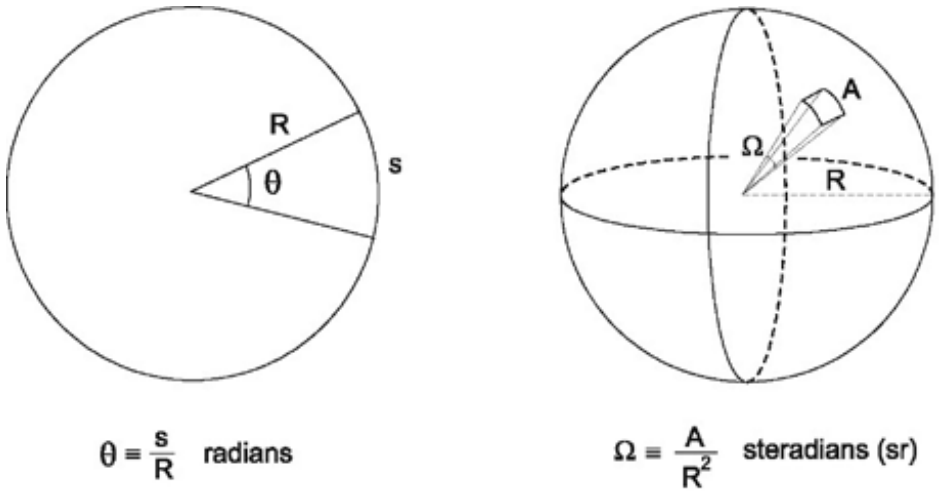


Figure 11 Illustration of an angle and a solid angle. Expressing the apparent size of a detector surface in steradians is the first step in reduction of the simulation complexity. Source: [44].

The solid angle of the detector surface is calculated using spherical trigonometry. This is a part of geometry, which deals with "triangles" drawn onto a spherical surface. Edges of these triangles are the shortest lines connecting two points on the surface of a sphere (orthodromes). A spherical triangle is shown in Figure 12.

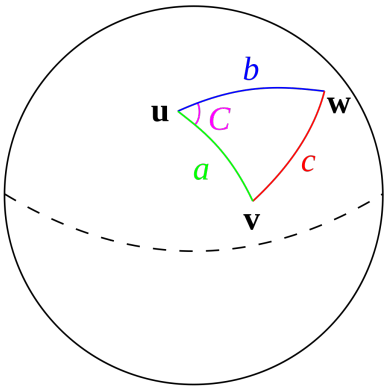


Figure 12 Illustration of an angle and a solid angle. Expressing the apparent size of a detector surface in steradians is the first step in reduction of the simulation complexity. Source: `https://commons.wikimedia.org/wiki/File:Law-of-haversines.svg`

The solid angle of a spherical triangle surface is equal to a spherical excess E of the triangle. The spherical excess is calculated as follows:

$$E = A + B + C - \pi \,, \tag{3}$$

where A, B, C are the angles between the orthodromes. To calculate these angles, a following equation derived from the *law of haversines* is used.

$$C = \mathrm{hav}^{-1}\left(\frac{\mathrm{hav}(c) - \mathrm{hav}(a-b)}{\sin(a)\sin(b)}\right). \tag{4}$$

The angles a, b, c represent central angles belonging to corresponding side of the triangle shown in Figure 12. This formula uses a trigonometric function *haversine* defined as:

$$\mathrm{hav}(\theta) = \sin^2\left(\frac{\theta}{2}\right) = \frac{1-\cos(\theta)}{2} \,. \tag{5}$$

A central angle θ between any two points on a spherical surface can be calculated from known coordinates of the center of the sphere (position of the radiation source) and the coordinates of the two points. The two points are projections of detector vertices on the spherical surface, however, the central angle remains the same if we use the coordinates of the vertices directly. The angle θ between vectors $\vec{u}, \vec{v}$ is calculated as

$$\theta = \cos^{-1}\left(\frac{\vec{u}\cdot\vec{v}}{\vec{u}\times\vec{v}}\right). \tag{6}$$

A solid angle Ω for the complete detector surface (consisting of up to 3 rectangles exposed to the radiation) is calculated by adding up the individual triangles. The solid angle of a complete spherical surface is equal to 4π steradians. Knowing the specific activity A of the radiation source, the apparent activity A_{app} of the detector is calculated as

$$A_{app} = A\frac{\Omega}{4\pi} \,. \tag{7}$$

The apparent activity serves as a frequency for the loop which simulates the photons. All these photons originate from the source, and are guaranteed to hit the detector surface. Since the detector is a cuboid, up to three sides can be facing the source. To determine, whether an oriented plane with normal vector is facing a point in 3D, the following inequality has to hold:

$$\vec{n}\cdot(\vec{s}-\vec{p}) > 0 \,, \tag{8}$$

where $\vec{n}$ is the normal vector of the plane, $\vec{p}$ is the coordinate vector of a point on the plane and $\vec{s}$ is the position of the source. With the exposed sides determined, a simulation step is performed. The procedure is described by Algorithm 1.

Algorithm 1 Radiation raytracer

```
1:  # get the number of rays to be simulated
2:  N ← ApparentActivity · ExposureTime · EnvironmentLoss
3:
4:  # simulate the rays
5:  procedure SimulationStep(Side, N)
6:      for N do
7:          p1 ← Side.RandomPoint
8:          r ← Raycast(source, p1)
9:          p2 ← Intersect(OtherSides, r)
10:         track ← Distance(p2 − p1)
11:         pe ← PhotoelAbsorptionProb(track, detector, source)
12:         if random(0,1) < pe then
13:             PhotonCounter ← PhotonCounter + 1          ▷ hit detected
14:         end if
15:     end for
16:     TimepixMessage ← PhotonCounter, ExposureTime
17:     ROS.Publish(TimepixMessage)
18: end procedure
```

Probability of photoelectric absorption by the detector depends on the length of track, energy of the photon and properties of the detector material. The absorption in obstacles is calculated in a similar fashion, based on the length of track, photon energy and material of the obstacle. However, a different coefficient is used, since the absorption can be caused by photoelectric effect but also by Compton scattering, pair production or bremsstrahlung.

The environment loss is calculated by casting a single ray to the center of the detector. It is the total product of absorption probabilities of all encountered obstacles, and also of the air. All other rays generated in the simulation step are ignoring obstacles and the air absorption completely. Instead, the environment loss is used to reduce their total count. This greatly reduces the time required to perform one iteration of the for loop.

To further reduce the number of necessary operations, the entire raytracing process is done in the coordinate frame of the detector, and the base vectors of the frame are equal to the depth, width and height of the detector.

With all the implementation steps in place, the plugin is able to simulate approximately 10000 ray hits per second with an average quad-core CPU. This performance is sufficient to simulate sources of radiation with activity in the order of GBq.

3.7 Visualization

A lightweight visualization library developed for this project is provided in package `geometry_visual_utils`. This package also contains implementations of all the geometrical calculations described earlier in this section. The visualization is aimed to be used with RVIZ (ROS Visualizer). This tool is independent on Gazebo, and is included in all currently supported ROS distributions.

Visualizing the radiation in RVIZ, rather than Gazebo itself, offers the option to run Gazebo without the GUI. This improves simulation performance on most computers, especially those without a dedicated graphics card. The difference between a world rendering in Gazebo and the same world in RVIZ is shown in Figure 13 and Figure 14. Gazebo implements a more powerful rendering engine, which allows for creation of more realistic scenes with textured object and advanced lighting. These properties are essential for projects that for example rely on visible-light cameras. RVIZ, on the other hand, offers by default a minimalistic visualization. The reduced hardware load is useful in projects, which strain the computer hardware by other calculations. Simulating a strong source of gamma rays definitely falls to the second category.

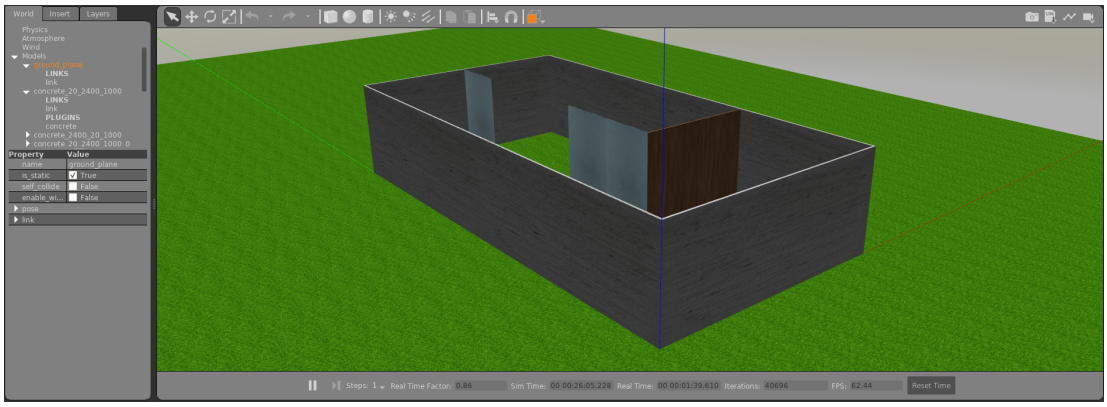


Figure 13 A scene in Gazebo rendered by OGRE3D. The models are textured and the user can use advanced lighting, shadows or shaders to achieve a more realistic look.

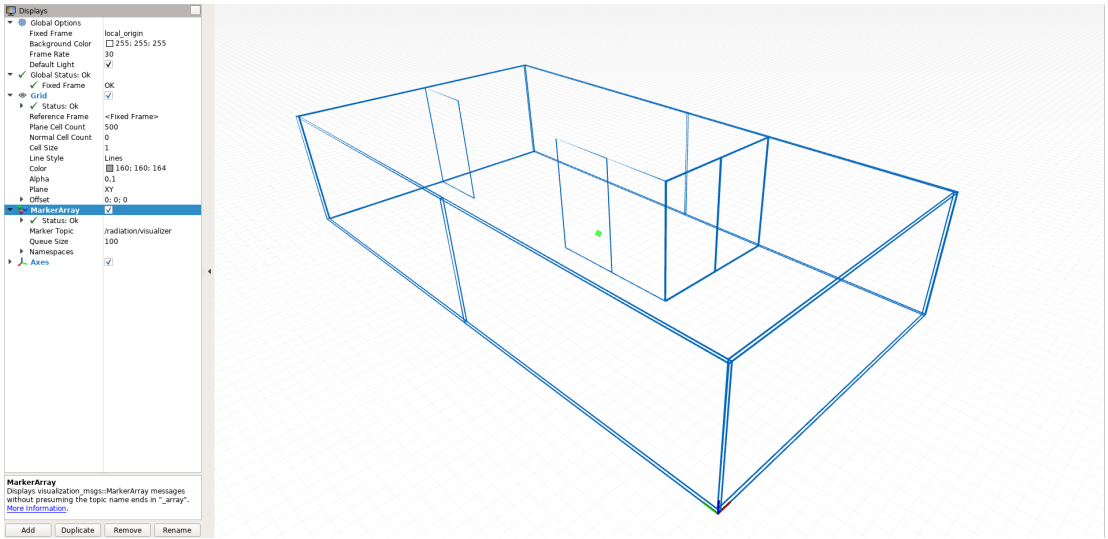


Figure 14 The same scene drawn in RVIZ using the custom library and lightweight visual markers to draw the outline of the geometry over the rest of the screen. The obstacles are outlined in blue and the source is visible through the walls as a green point.

The tool uses RVIZ *markers*, which are 2D geometric primitives drawn over the scene, such as points and line segments. These markers are used to outline the radiation obstacles, the Timepix detector, and to highlight individual rays generated by the raytracer.

The primitives can be added by using methods `addPoint, addRay, addRectangle` and `addCuboid`. Each method takes the following arguments: an appropriate geometric object (defined in the header file `geometry_utils.h`), colors in RGB[7] and a line width in meters. All objects are drawn in a batch by calling the method `publish`, and erased by the method `clear`.

Behavior of the Timepix plugin can be easily illustrated by the visualization tool. Figure 15 shows the Timepix detector bombarded by photons from two radiation sources. Both sources are samples of ^{137}Cs with the same specific activity, and the same distance from the detector. One source is placed in front of the largest detector face, while the other source is hitting mostly the side face. The number of rays projected by each source demonstrates, that the apparent area for the second source is clearly smaller. However, the distance which the photons travel through the detector is larger for the side-hitting radiation. This increases the chance of photoelectric absorption of the photons, and therefore also the chance of detection.

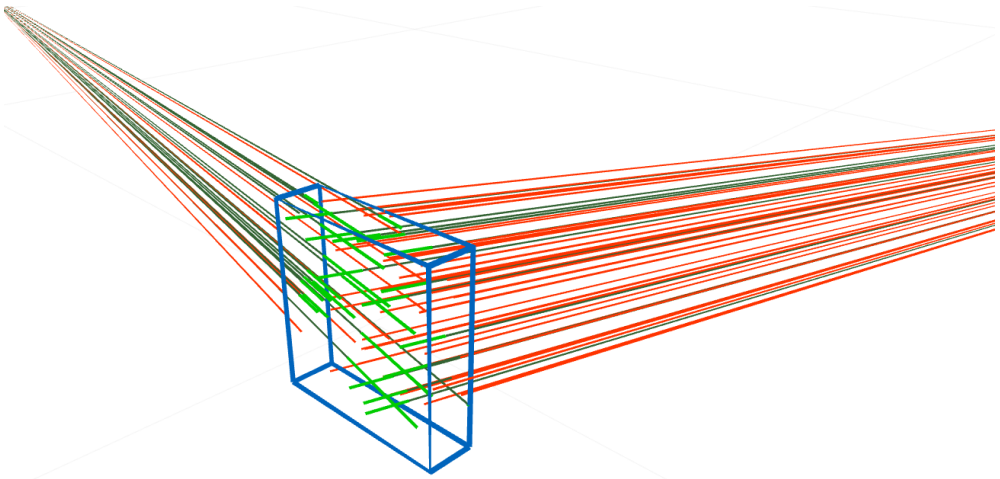


Figure 15 A visualization of the raytracing procedure drawn in RVIZ. The detector is shown in blue, and the thickness of the detector is exaggerated. Two samples of Cesium-137 are hitting the detector with gamma photons, which are represented by red and green lines. One source is mostly hitting the largest face, and the other is mostly hitting a smaller side face. Photons, which are absorbed by a photoelectric effect and detected, are assigned dark green color. The distance, which these photons travel through the detector is highlighted in bright green. Red color is assigned to photons, which pass through the detector unnoticed.

[7] Red, green, blue

4 Source localization

This chapter is focused on implementation of algorithms, which control a group of UAVs and estimate the position of a radiation source, based on onboard measurements. The goal was to design a flexible system, which does not rely on a specific number of UAVs or a specific layout of the obstacles.

This thesis presents two approaches to the localization problem. A mapping approach, which relies on the UAVs following a predefined path and taking continuous measurements. The output of this algorithm is a map of radiation intensity. By finding a global maximum in the map, the position of the source can be determined.

For larger maps, the mapping task can get very time-consuming. Therefore, a second approach solves the problem by actively searching for the source. This method utilizes the movement of UAVs, to enhance the information gained from the Timepix sensor. This way, the direction of incoming gamma rays can be estimated. Multiple estimates provided by all active UAVs are then fused into one with the use of a Kalman filter.

The implementation allows for use of a group of UAVs of arbitrary size. The minimal required number of UAVs is one, the upper limit is not specified. However, the available computational power and the amount of free space in the environment should be taken into consideration. All employed UAVs are considered equally capable of fulfilling this task. Therefore, adding more UAVs does not immediately guarantee more accurate results. The main advantage of using a group of UAVs over a single UAV resides in reduced time requirements.

4.1 Assumptions

A map of the whole scene, where the localization takes place, is known in advance. It is provided in the form of an occupancy grid. The size and resolution of the grid are also known in advance.

All obstacles in the scene are static, and so is the radiation source. In all scenarios presented in this thesis, there is exactly one radiation source placed in the scene. Neither the activity of the source, nor the material is known in advance. The source may be placed in an arbitrary *free* position (not inside a wall) anywhere in the scene. However, the position has to be reachable by the UAVs.

No sensor noise or radiation background is simulated. The UAVs continually communicate with a ROS central node, and also with each other. The communication is not obstructed by obstacles or the ionizing radiation. It is also assumed, that the position of all UAVs in the world coordinate frame is known.

4.2 Mapping

In a known map, the task is to find a space-filling path, which sufficiently covers the reachable area. Velocity of the UAVs, and the exposure time of Timepix detectors has to be chosen in such way, that the measurements are available with satisfactory density. In the case of mapping, there is a trade-off between the map detail and required time. In a multi-UAV scenario, the path is divided into segments of equal length and each segment is assigned to one UAV.

In this work, the generated path follows a "lawn mower" pattern. An example of the path is shown in Figure 16. Each UAV publishes the Timepix readings to a central node, that generates a radiation intensity map. The map is derived from the occupancy grid, and shares the same size and resolution. In fact, two maps are created. One map contains raw data, i.e., the total particle count for each grid cell. For the second map, the node also measures time spent over each cell. Dividing the raw particle count by the time produces a map compensated for deviations from a constant velocity of the UAV. These deviations may be caused by turning or, in the case of outdoor experiments, external factors such as wind gusts. In the second map, the cell values directly represent the measured apparent activity in Becquerels.

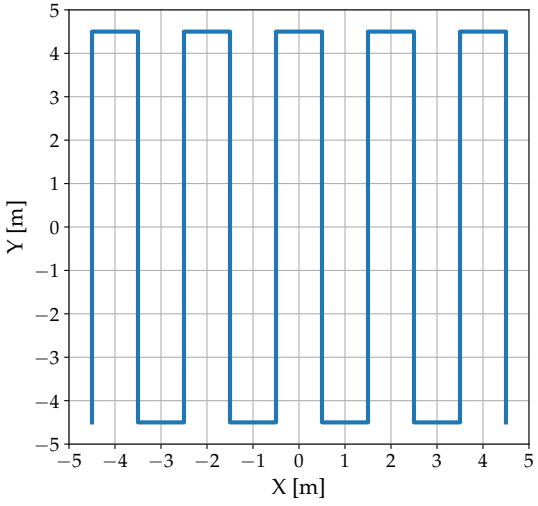


Figure 16 An example of a mapping flight path. The goal is to create a map of radiation in all reachable cells of the map grid. The UAV maintains a constant velocity and height above ground, while it follows this path. The radiation map is constantly updated with new Timepix measurements.

4.3 Active search

Symbol	Description
n	Number of system states
m	Number of system inputs
$\mathbf{x}_k$	System state vector at time k
$\hat{\mathbf{x}}_k$	Estimated state vector at time k
$\hat{\mathbf{x}}_{k+1\|k}$	Estimate based on data from time k
$\hat{\mathbf{x}}_{k\|k}$	Estimate corrected by current measurement
$\mathbf{y}_k$	State observation vector
$\tilde{\mathbf{y}}$	State residual (difference between prediction and measurement)
$\mathbf{I}$	Identity matrix
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$	State space matrices
$\mathbf{K}$	Kalman gain
$\mathbf{Q}$	Process covariance
$\mathbf{R}$	Observation (measurement) covariance
$\mathbf{P}$	Estimation covariance (uncertainty of the estimate)
$\vec{s}$	Coordinates of a radiation source in world frame

Table 1 A listing of all symbols and variables used in this section.

4.3.1 Control node

The highest level of control is performed by a central ROS node. The node collects radiation measurements and odometry, issues new commands to the UAVs, and performs real time estimation of position of the source. The control node is implemented as a state machine, which tracks the state of every UAV separately. A diagram of the control node operation is shown in Figure 17. Every active UAV can occur in one of following four states:

- pathfinding,
- repositioning,
- direction estimation,
- data fusion.

The node continuously iterates over all active UAVs, and issues them commands based on their current state. The control loop performs 100 updates per second, since it is also the rate, at which the UAVs publish their odometry updates. The exposure time of Timepix models was set to 0.1 seconds. Hence, the radiation measurements are only available once every 10 odometry updates. For calculations, which require both odometry and radiation measurement, the system skips the current UAV until both measurements are available.

The exposure time was chosen empirically, based on the results of real experiments (presented later in Section 5.1) and simulations (Section 5.2). The value represents a compromise between quality of measurement and localization speed. Longer exposures would allow the UAV to detect more photons at larger distances. However, the process would also require more time. As with many applications of UAVs, the time constraint is induced mainly by the endurance of an onboard battery.

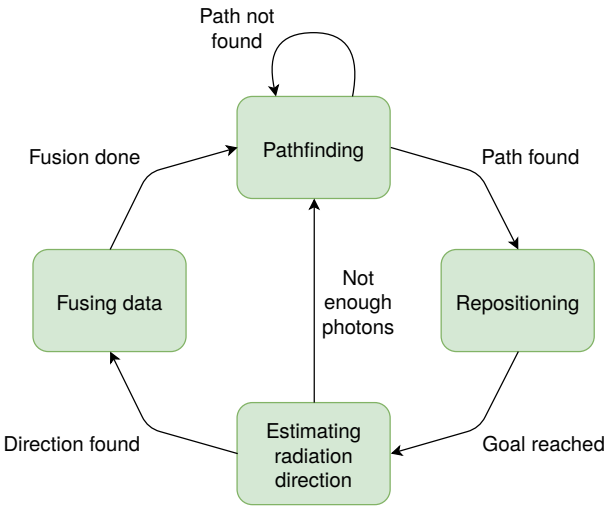


Figure 17 A diagram showing the operation of the state machine which controls the UAV group. Every UAV operates independently on the others, and the operation repeats in cycles. In every cycle, the UAV moves to a new position in the map, performs measurements and updates the estimated position of the radiation source. If the UAV does not detect enough photons, the direction cannot be estimated and the data fusion step is skipped.

4.3.2 Estimating radiation direction

In case of heavier ions or neutrons, the direction of incoming radiation can be estimated directly from shapes of particle tracks in the Timepix image. However, gamma photons leave tracks of only one or two pixels in size. Therefore, the sensor can only provide information about radiation intensity.

As was established previously, the apparent intensity depends on the surface area of the detector exposed to the source. Moving the detector relative to the source will cause changes in measured intensity. The source remains static, therefore by changing the position of the UAV and taking continuous measurements, an intensity gradient can be determined. Afterwards, a gradient ascent algorithm can be employed, to find the position of the source.

This approach has been proven to work well in open environments without obstacles, as was presented in thesis [45]. However, it is not applicable in an environment with obstacles, which attenuate the radiation. With obstacles, the gradient function is no longer monotonous, and the algorithm is only capable of reaching local extrema.

Another approach is to rotate the detector around one of its axes. Two counteracting physical principles affect the amount of detected photons. On one hand, the apparent area of the detector changes, as illustrated by Figure 18. The highest number of photons hits the detector if the largest face is perpendicular to the rays projected by the source.

On the other hand, the photons travel the shortest distance through the detector in this orientation. This decreases the chance of a photoelectric absorption. For the shortest track in a 300 µm Silicon chip, the chance of absorption (and subsequent detection) is only 1.61% in the case of 662 keV photons emitted by ^{137}Cs, and 2.03% for the 60 keV photons emitted by ^{241}Am. Simulation results shown in Figure 19 demonstrate, that the two principles nearly balance each other out for all but one very specific case. This case occurs, when the largest face of the detector is parallel to the incoming gamma rays. This causes a significant decrease in measured intensity. Based on this observation, it was decided to search for the direction with the lowest apparent

activity instead, and then rotate the direction by $\frac{\pi}{2}$ radians.

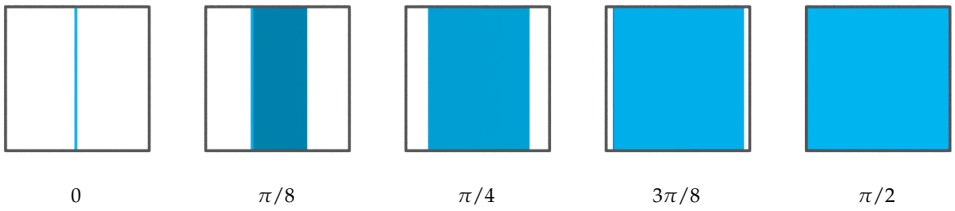


Figure 18 An illustration showing the changes in apparent detector area (blue) exposed to the radiation, when the detector is rotated. The illustration shows the detector from the point of view of the source. The labels represent angle between the gamma rays and the largest detector face. The changes in apparent area result in changes in measured radiation intensity. The most significant decrease in intensity corresponds to parallel orientation to the gamma rays (far left image).

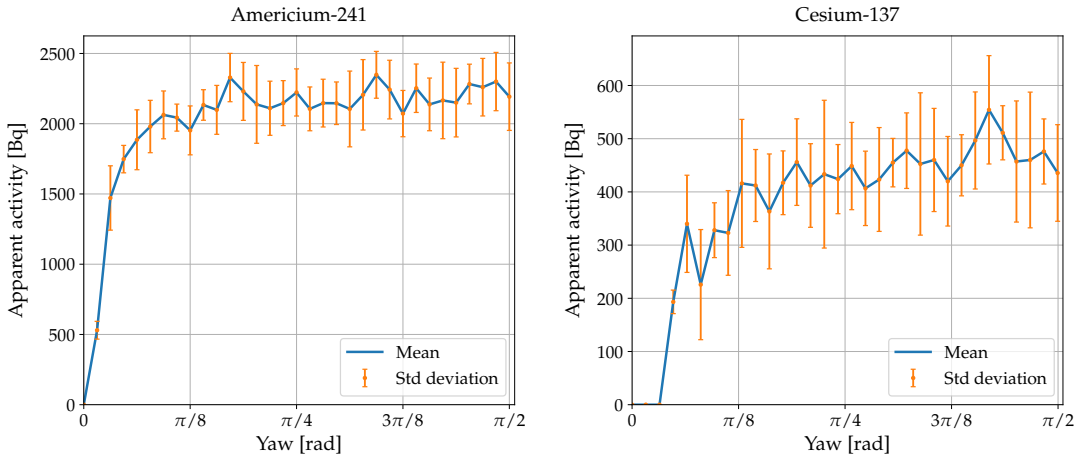


Figure 19 A simulation showing the apparent activity of a radiation source in relation to the orientation of the detector. The detector was positioned at coordinates $[0, 0]$ with the largest face perpendicular to axis X. The radiation source was placed at coordinates $[0, 5]$. Both objects were placed to the same height. The simulation was performed for two different sources of radiation, both with specific activity of 2 GBq. Interesting is the lack of a distinct maximum in the intensity, while the minimum is apparent for both sources.

On these foundations, a following control algorithm has been designed. The UAV maintains a constant position. When a new Timepix message is received, the UAV increases its yaw by a previously specified amount. The increment was set to $\frac{\pi}{20}$ as a compromise between angular resolution and require time. This process is repeated until the interval $\langle 0; \pi \rangle$ radians is covered. Afterwards, the direction with the lowest photon count is rotated by $\frac{\pi}{2}$ radians. Under this angle, a line is projected from the position of the UAV. The line is cut into two segments – forward and backward. The algorithm only uses the segment, which points in the direction of a previous source estimate. This direction estimate is interpreted as a measurement in the next step.

4.3.3 Data fusion

This section faces a challenge of translating the direction estimate into a position estimate. The solution was implemented in the form of a Linear Kalman filter (LKF), which takes the measurements done by all UAVs as the input.

Kalman filter

Kalman filter is an algorithm for estimating the state of a dynamic system, based on the knowledge of a previous state and a new measurement. The principle was first introduced in [46]. Since then, variations such as the Extended Kalman filter (EKF) [47, 48] and an Unscented Kalman filter (UKF) [49, 50] were developed for use with non-linear systems. The algorithm uses a dynamic system model, and assumes that the input data, internal state and measurements contain statistical noise.

The filtering process is divided into two steps – prediction and update. In the prediction step, a new state of the system is predicted, based on the previous state and the model of the system. In the correction step, the prediction is compared with a new observation (measurement), and the state estimate and its covariance are updated.

Let us have a linear time-invariant (LTI) system with state matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$, state vector $\mathbf{x}$, input vector $\mathbf{u}$ and output vector $\mathbf{y}$. The system can be described by following difference equations:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k + \mathbf{w}_k \ , \tag{9}$$

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + \mathbf{D}\mathbf{u}_k + \mathbf{v}_k \ . \tag{10}$$

The system contains a process noise $\mathbf{w}$ and an observation noise $\mathbf{v}$, which encompass all inaccuracies of the model, errors introduced by the measurement method, sensor noise etc. The noise variables are assumed to be uncorrelated, with a Gaussian probability distribution:

$$p(\mathbf{w}) \sim \mathcal{N}(0, \mathbf{Q}) \ , \tag{11}$$

$$p(\mathbf{v}) \sim \mathcal{N}(0, \mathbf{R}) \ . \tag{12}$$

Here, $\mathbf{Q}, \mathbf{R}$ represent corresponding covariance matrices of the noise. The filtering process can be formally written as a sequence of linear expressions. Firstly the prediction step, which uses the state estimate and input from a previous step:

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{A}\hat{\mathbf{x}}_{k|k} + \mathbf{B}\mathbf{u}_k \ , \tag{13}$$

$$\mathbf{P}_{k+1|k} = \mathbf{A}\mathbf{P}_{k|k}\mathbf{A}^T + \mathbf{Q}_k \ . \tag{14}$$

The $\hat{\cdot}$ operator denotes, that the vector represents only an estimate rather than the actual value. The matrix $\mathbf{P}$ represents a covariance of the estimation error. The newest prediction is updated if a new measurement is available. The update phase begins by calculating residual $\tilde{\mathbf{y}}$, which is the difference between a predicted output and an actual measurement. Afterwards, a Kalman gain $\mathbf{K}$ is calculated and the state estimate is updated. Finally, the estimate error covariance $\mathbf{P}$ is also updated. Mathematically, the update phase can be expressed by the following equations:

$$\tilde{\mathbf{y}}_k = \mathbf{y}_k - \mathbf{C}\hat{\mathbf{x}}_{k|k-1} \ , \tag{15}$$

$$\mathbf{K} = \mathbf{P}_{k|k-1}\mathbf{C}^T \left(\mathbf{C}\mathbf{P}_{k|k-1}\mathbf{C}^T + \mathbf{R}\right)^{-1} \ , \tag{16}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}\tilde{\mathbf{y}}_k \ , \tag{17}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}\mathbf{C})\,\mathbf{P}_{k|k-1} \ . \tag{18}$$

Due to the recursive nature of the filter (new estimate only depends on data from the previous step), the computation does not require a lot of memory. Moreover, the prediction and update steps for LKF are represented as a series of linear expressions. This means, that implementation is fairly simple, and modern computers can often perform the filtering in real-time.

LKF is an *optimal* state estimator, if the process noise and observation noise are uncorrelated and Gaussian white[1], the exact system model is known, or the exact properties of the noise are known. If any of these conditions is met, Kalman filter *minimizes* the mean square error of the estimation. In other cases, the estimation is only suboptimal, but may still provide a lower mean square error than individual measurements [48, 51].

Source position estimation

As was described earlier, a single UAV is capable of estimating the direction of incoming ionizing radiation, by changing yaw while hovering in one place. However, it is impossible to determine the distance between the UAV and the radiation source, if the activity of the source is unknown. There is no way to distinguish a faint radiation source located near the helicopter from a strong radiation source in a distance. Therefore, at least two measurements from different locations are required.

The implementation considers all UAVs equally capable of determining the direction. All measurements are passed to the control node, which fuses them into one estimation via LKF. This provides a lot of flexibility, as the system is completely agnostic to the UAV group size. A single UAV can perform multiple measurements and estimate the source position just as well as a group of UAVs. Nevertheless, a group of UAVs finishes the search much faster, due to multiple measurements being taken simultaneously.

The position of each UAV is assumed to be known in all steps of the filtering. Therefore, the only unknown variables in the system are the world frame coordinates of the radiation source. The direction estimate is not perfectly accurate, since the UAV performs discrete changes in yaw. For this reason, the estimation cannot be simply done by finding an intersection of two lines projected in the direction of incoming radiation. Even small inaccuracies in the direction estimate result in an estimation error, which grows with increasing distance.

This is where the Kalman filter can be used very efficiently. A line projected by a direction estimate can be imagined as position estimate with a very large uncertainty in this direction. This analogy is illustrated in Figure 20. The uncertainty in the remaining directions is calculated as the shortest distance between a previously estimated position and the projected line.

[1] Random signal with a constant intensity across all frequencies, following a normal distribution in time domain with a zero mean.

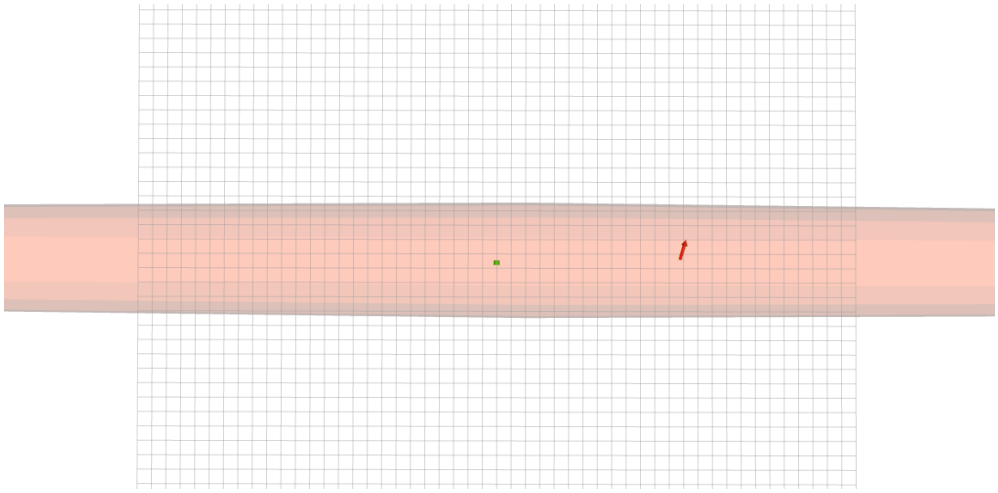


Figure 20 An RVIZ illustration of how a direction estimate is represented in the system. The position and yaw of the UAV is represented by a red arrow. The true position of the source, which the UAV is supposed to determine, is marked by a green dot. The orange ellipse represents the estimation uncertainty, which is extremely elongated in the direction of the estimate. Intensity of the source is unknown, therefore the UAV cannot determine, whether the radiation originates from a weak source nearby, or a strong source in a distance. The size of each grid cell is 1 m^2.

Firstly, the state space model for the localization problem has to be specified. The state vector $\mathbf{x}$ only represents the unknown coordinates of the radiation source:

$$\mathbf{x} = \vec{s} = \begin{pmatrix} s_x \\ s_y \\ s_z \end{pmatrix} . \tag{19}$$

The source does not move, and is not affected by outside forces. Therefore, the system does not contain any input and the state matrices are reduced to:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{D} = 0 \ . \tag{20}$$

With this specific system, the prediction step of the Kalman filter only increases the estimation covariance:

$$\hat{\mathbf{x}}_{k+1|k} = \hat{\mathbf{x}}_{k|k} \ , \tag{21}$$

$$\mathbf{P}_{k+1|k} = \mathbf{P}_{k|k} + \mathbf{Q} \ . \tag{22}$$

The direction estimate is introduced as a new state observation. The observation covariance is first calculated using a *camera coordinate frame*, which is illustrated in Figure 21. The Timepix serves as the camera for this frame. In the camera coordinates, the uncertainty of direction estimate increases the covariance in the direction of X axis. The covariance in the direction of Z axis is set to an extremely large value, since the distance between the source and the detector is unknown.

After assembling the covariance matrix, it is transformed back into the world coordinate system. The transformation between a world frame and the UAV frame is known in advance from the odometry messages published by the UAV. Transforming the camera coordinates into the UAV frame requires a series of rotations about the coordinate frame origin to correctly align the axes (X – forward, Z – up).

The position estimate and its covariance in the world coordinate frame represent a measurement $\mathbf{y}_k$ for the Kalman filter. The update step for this specific system is described by following equations:

$$\tilde{\mathbf{y}}_k = \mathbf{y}_k - \hat{\mathbf{x}}_k \ , \tag{23}$$

$$\mathbf{K} = \mathbf{P}_{k|k-1} \left(\mathbf{P}_{k|k-1} + \mathbf{R} \right)^{-1} \ , \tag{24}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}\tilde{\mathbf{y}}_k \ , \tag{25}$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{KC})\, \mathbf{P}_{k|k-1} \ . \tag{26}$$

After the update step of the Kalman filter, the UAV is commanded to move to a new position and take a new measurement.

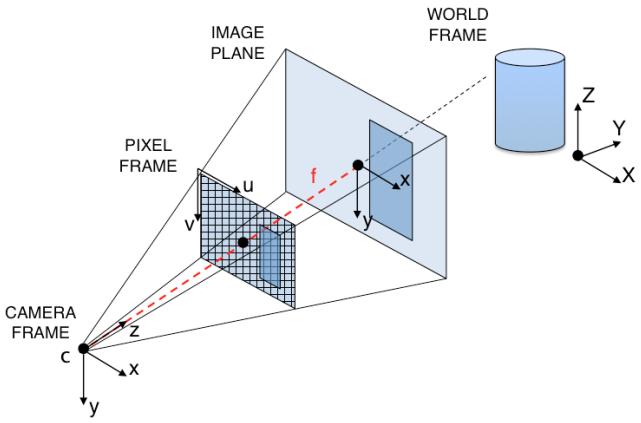


Figure 21 This image shows the relation between a camera coordinate frame and the world frame. The covariance of every direction measurement is calculated in these coordinates. In this frame, it can be easily expressed, that the distance between a source and the detector is unknown. The measurement covariance needs to be elongated in the direction of Z axis. Image source: `http://people.bu.edu/chenyua/CS585finalproject/2_cameraCoords.png`

4.3.4 Pathfinding

The path planning is done in the provided occupancy grid describing the world. In the grid, obstacles are expanded by twice the diameter of a UAV. This ensures that the UAV will maintain a safe distance from obstacles. The expansion also provides a room for deviations, which may be introduced by the self-localization methods.

A* algorithm

The control node uses a graph-searching algorithm A* to find the shortest collision-free path to a goal position. It belongs to a group of *informed* search algorithms, i.e., the start and the end positions are known in advance. The graph is represented by the occupancy grid, and cells of the grid represent nodes of the graph. Each cell is assigned a cost based on a cost function:

$$f(c) = g(c) + h(c) \ , \tag{27}$$

where $f(c)$ represents the cost of cell c, $g(c)$ is the total cost of path from start to cell c and $h(c)$ is a heuristic estimate of the remaining cost of path from a current cell to the end. An optimal path is a sequence of cells, which minimizes this cost function.

The process of finding a path using A* is illustrated by Algorithm 2. The algorithm is widely used in robotics as well as in the video game industry.

In the case of this work, all transitions between two cells neighboring cells are equal in cost. The heuristics is calculated as the shortest straight distance between a current cell and the end point, disregarding the obstacles. The evaluation uses Euclidean metric, which defines the distance d between grid cells $\mathbf{c}_1, \mathbf{c}_2$ with coordinates x, y as:

$$d(\mathbf{c}_1, \mathbf{c}_2) = \sqrt{(c_{1x} - c_{2x})^2 + (c_{1y} - c_{2y})^2} \,. \tag{28}$$

The next goal position for each UAV is generated randomly. The goal may be any free cell in the grid, which is at least 8 cells away from the current position. If a path towards the goal does not exist, new goals are generated until a path is found.

Algorithm 2 A* planner

```
 1: procedure FindPath(grid, start, end)
 2:     open ← PriorityQueue                          ▷ highest priority = lowest cost
 3:     closed ← List
 4:     open.Add(start)
 5:     while not open.Empty do
 6:         current ← open.RemoveFirst
 7:         closed.Add(current)
 8:         if current.Equals(end) then                ▷ goal reached → build path
 9:             while not current.Equals(start) do
10:                 path.Add(current)
11:                 current ← current.Parent
12:             end while
13:             return path
14:         else
15:             neighbors ← current.Expand
16:             for n in neighbors do
17:                 n.AssignCost
18:                 if closed.Contains(n) then
19:                     if n.Cost < closed.Get(n).Cost then
20:                         closed.remove(n)          ▷ node rediscovered from a better path
21:                     else
22:                         continue                  ▷ do not reopen closed nodes
23:                     end if
24:                 end if
25:                 if open.Contains(n) then
26:                     if n.Cost < open.Get(n).Cost then
27:                         open.Remove(n)            ▷ found a better path for the node
28:                     else
29:                         continue
30:                     end if
31:                 end if
32:                 open.Add(n)
33:             end for
34:         end if
35:     end while
36: end procedure
```

Repositioning

The UAVs of the Multi-Robot Systems group implement a layered control structure. The lowest layer is represented by the Pixhawk[2] – an off-the-shelf flight controller with an embedded IMU[3]. This layer handles the stabilization of yaw, pitch and roll and controls the speed of all motors. This layer also accepts control inputs from the higher layers in the form of a desired UAV orientation rate and the total motor thrust.

Above the Pixhawk is an SO(3)[4] non-linear state feedback controller [52]. This layer controls the angular rate of the UAV. This controller allows the UAV to perform fast and aggressive maneuvers, however it requires a smooth and feasible input. The input is provided in the form of a desired UAV state vector, consisting of a position, velocity and acceleration vectors.

The highest layer of the system consists of a Model Predictive Control Tracker (MPC Tracker) [53]. This layer takes the desired coordinates of the UAV as inputs. It utilizes an LTI model, which represents the transition of the input into the state vector of the UAV, and predicts the next system states in a predefined time horizon. The tracker produces a smooth and feasible reference for the SO(3) controller. It also communicates with other UAVs, which use the same system, to prevent mutual collisions. In case of a collision of two predicted trajectories, the tracker performs a collision avoidance maneuver by changing the trajectory. The entire control structure is fairly complex, and has been continually refined over the course of several years. A much more in-depth description of the control structure is provided in [17, 54, 55].

The output of the A* algorithm serves as a fourth control layer. It provides a sequence of waypoints, which are passed to the MPC controller as a desired trajectory. The planner smooths out the trajectory and handles the collision avoidance. To maintain measurement unbiasedness among the UAVs, all of them operate at the same height above ground.

In case of overlapping trajectories, the collision avoidance intervenes by increasing the height above ground for one of the UAVs. During the direction estimation phase, the UAV disables the avoidance system and acts as an obstacle for the other UAVs. This ensures, that the direction estimation is never interrupted by another UAV passing nearby.

[2] `http://pixhawk.org/`

[3] Inertial measurement unit

[4] Spherical orthogonal subspace of a 3D Euclidean space. The space of all rotations preserving coordinates of the origin.

5 Evaluation

This chapter demonstrates the functionality of the newly developed Gazebo plugins presented in Chapter 3 and the mapping and localization algorithms presented in Chapter 4. The evaluation was performed in a simulated environment of Gazebo and also in real-world experiments, which included an actual radioactive source and one UAV equipped with a Timepix detector.

In the first section, results of the real experiment are presented. This experiment served as a proof of concept, and a valuable source of data for further research. It demonstrates, that a Timepix can be mounted onto a UAV, and used to detect even a faint source of radiation in an obstacle-free environment. Afterwards, the results are compared to simulations to verify the behavior of the newly implemented Gazebo plugins. Finally, the localization methods introduced in Sections 4.2, 4.3 are tested in a more complex 3D environment with obstacles.

5.1 Baseline experiments

A series of experiments was conducted in a real outdoor environment. The results obtained during the experiments serve as a base for all software implementation, which was described in previous chapters. The main goals of the experiments were to determine:

- whether a UAV equipped with a Timepix is even capable of providing useful data,
- how are the measurements affected by the height above ground,
- what exposure time should be used for the Timepix readout,
- which issues need to be overcome before deployment in a real-world application.

5.1.1 Preconditions

The experiments were performed in a restricted compound of the Czech National Institute for Nuclear, Chemical and Biological Protection[1]. The flights were performed outdoors, over a flat concrete surface. The UAV used for this task was a DJI F450 quadrotor of the Multi-Robot Systems group, which is shown in Figure 22. The UAV is equipped with a Pixhawk Mini flight controller, uBlox 3DR GPS with a compass, downward facing monocular camera BlueFOX and a downward facing laser rangefinder Garmin LIDAR-Lite v3 which measures the height above terrain. The UAV also carries an onboard computer Intel NUCi7, equipped with a quad-core CPU, 8 GB RAM and a solid state drive (SSD). The computer is running operating system Ubuntu 18.04 with ROS in version Melodic.

A Timepix detector with a 300 µm Silicon chip was used as the radiation detector. It was connected to a USB port of the onboard computer via an interface called FitPIX. The detector was mounted in a 3D printed holder below the UAV, with the exposed face of the detector facing forward, in the direction of X-axis in the coordinate frame of the helicopter. The mount is illustrated in Figure 22. The exposure time was chosen empirically, by placing the UAV and the radiation source two meters apart,

[1] `http://www.sujchbo.cz/`

and increasing the exposure time until multiple photons were observed in the Timepix images. The entire communication over ROS topics, including the images from Timepix and the BlueFOX camera, were recorded into a rosbag file for further processing.

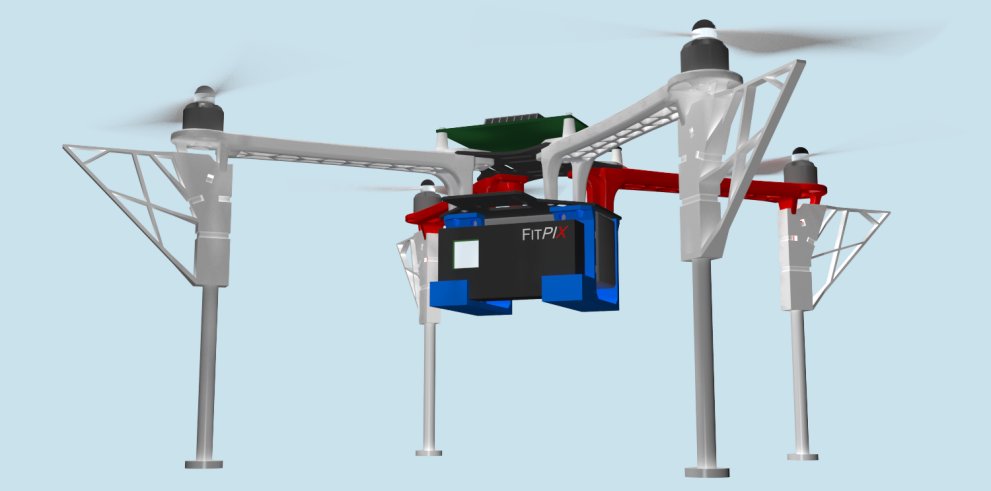


Figure 22 A rendering of the DJI F450 with a custom holder for the Timepix sensor. The holder is shown in blue color. It is mounted to the bottom central plate of the UAV, with the Timepix chip (silver) pointed forward. This holder was specifically designed to carry Timepix connected to the USB interface FitPIX.

The UAV was fusing optic flow in the image from the downward facing BlueFOX camera, and the GPS information to provide ground truth for the radiation mapping. Heading of the UAV was obtained from the uBlox compass.

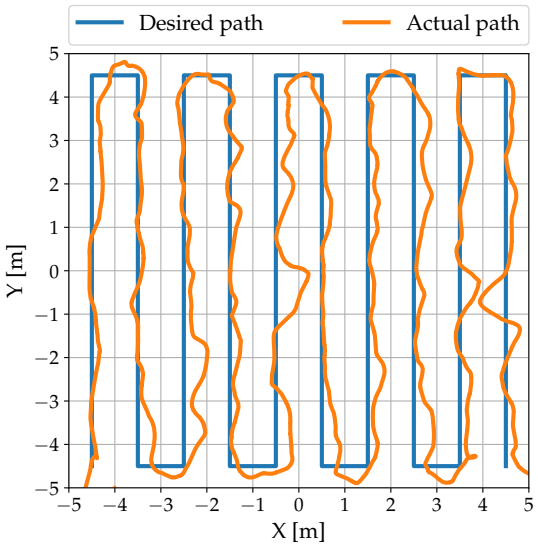


Figure 23 A “lawn mower” flight path used during the experiments. The UAV maintained a constant height above ground during the flight and the top speed was limited to 0.5 meters per second. Reducing the movement speed allowed the Timepix to collect more data over each grid cell.

The radiation source used in the experiments was a sample of Americium-241 with a specific activity of 500 MBq. The source was placed in a known position with local

position $[0, 0, 0]$ in most of the experiments. The origin of the local coordinate system was reset to match the starting position of the UAV for every flight. The UAV autonomously followed a predefined flight path shown in Figure 23, and covered a square area between <-5,5> in X and Y of the local coordinate system. The horizontal velocity of the UAV was limited to 0.5 meters per second.

The unique opportunity to perform the experiments was limited to only one day, and the flights had to be performed on the premises of another institution. Therefore, it was impossible to wait for more hospitable weather conditions, or to introduce significant changes to the hardware and software of the UAV. The plan was to perform indoor experiments as well, but due to a poor indoor performance of the onboard compass, the experiments had to be postponed. In the future, the compass will be replaced by optic flow or a rotating LiDAR sensor to determine the yaw.

Despite minor setbacks, the experiments still yielded extremely valuable results. The author is deeply grateful to all people and institutions involved in creating this opportunity.

5.1.2 Results

The recorded rosbag files were processed offline after the experiments. The full images provided by Timepix were analyzed by a pattern recognition algorithm provided with the Rospix package. The pattern recognition proved to be working in real-time when playing the rosbag at original speed. Therefore, in future missions the processing can be done directly by the onboard computer.

Since the emission type of the source was known (only gamma rays), the pattern recognition was able to separate the photon tracks originating from the ^{241}Am from the natural radiation background[2]. The amount of events detected in every image was paired with odometry of the UAV, and assigned to a cell in a grid by the mapping ROS node. The size of each cell was set to 1 m^2.

The strength of the radiation background measured during one of the flights is shown in Figure 24. Even though the background was filtered out by the pattern recognition software, it reduced the total amount of photons which could be detected. In the Timepix image, each track produced by the background radiation reduced the effective area of the detector, which could be used to capture the gamma rays.

Five flights were performed, each with slightly different parameters. The variations are listed in Table 2.

#	UAV height above ground [m]	Source X,Y,Z position [m]	UAV heading
1	1.5	[-2, 0, 0]	Aligned with X axis
2	1.5	[0, 0, 0]	Aligned with X axis
3	3.0	[0, 0, 0]	Aligned with X axis
4	0.7	[0, 0, 0]	Aligned with X axis
5	1.5	[0, 0, 0]	Always facing the source

Table 2 The schedule of real experiments and the variations in parameters between individual flights. The goal was to determine a suitable height for future mapping missions, and determine whether the heading of the UAV affects the map quality in any way.

[2]Naturally occurring radioactive elements, cosmic rays, solar radiation

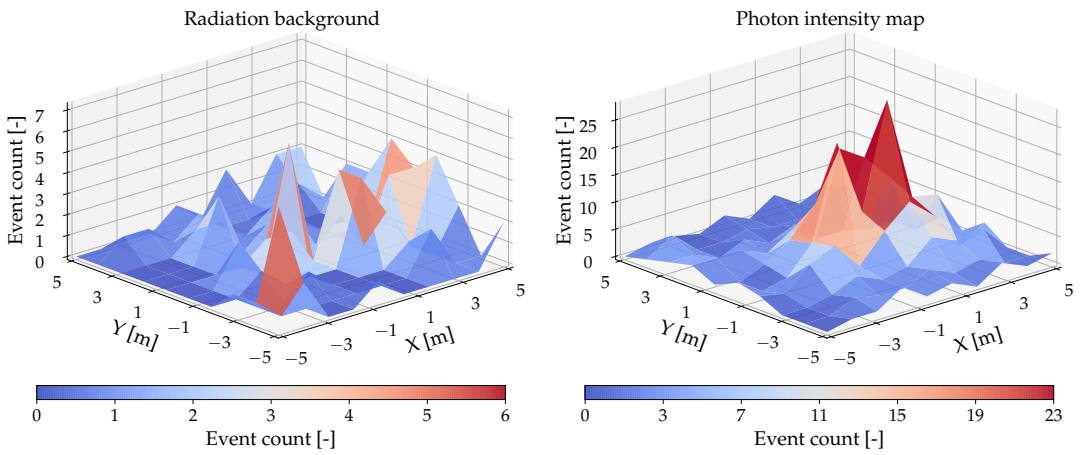


Figure 24 Map of ionizing particles, which did not originate directly from the source, is shown on the left. These events include high energy electrons and heavier ions caused by cosmic rays and naturally occurring radioactive elements, and form a radiation background. These particles were filtered out by an image processing algorithm, to produce a map of photons as shown on the right. Since the Americium sample was by far the strongest source of gamma photons in the area, this map can be used to determine its position.

The UAV had to endure gusts of wind, illumination changes from broad daylight to overcast, and also light rain. These challenging conditions caused unpredictable deviations from a desired flight path (as illustrated in Figure 23), changes in UAV velocity and also affected the radiation measurements.

For these reasons, all further maps presented this section are in the normalized form, showing a measured activity in Becquerels. The maps from individual flights, after photon-only filtering, are presented in Figures 25, 26, 27, 28, 29.

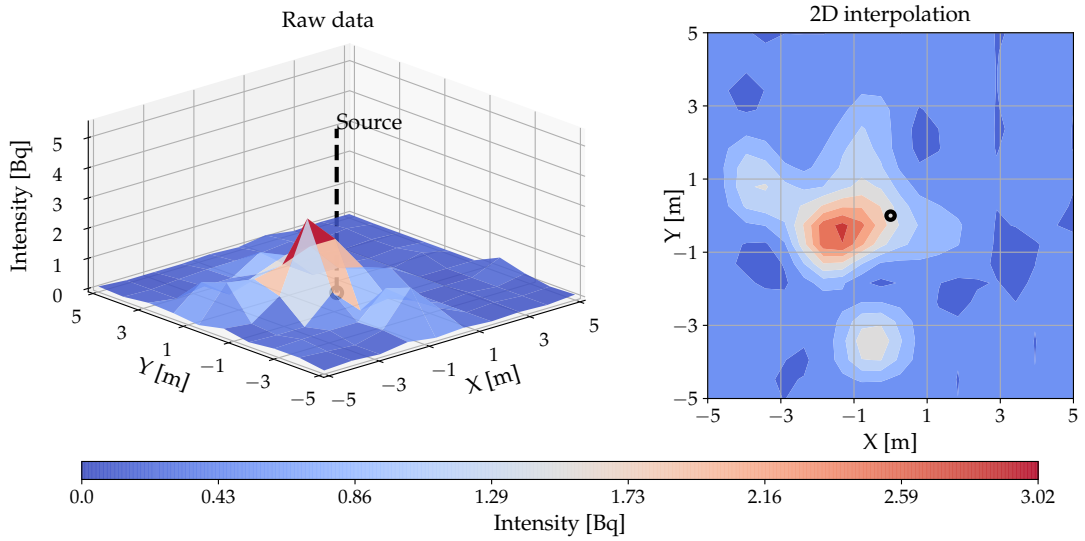


Figure 25 Radiation map created from the measurements obtained during the experiment #1 from the height of 1.5 m. Actual position of the source is highlighted by a black marker. During this experiment, the source was not placed at the origin of the local coordinate system. Instead, it was located in position $[-2, 0, 0]$ m.

Due to the small size of searched area and large size of individual grid cells, the resulting maps have a very coarse resolution. A third order spline interpolation was used on the grid to produce a radiation intensity map with a finer resolution. The approximation was done using a Python image processing library `scipy.ndimage`.

Since the sensor is pointed forward (in the direction of the X coordinate in frame of the UAV), the measured intensity tends to decrease, if the UAV is located directly overhead the radiation source. The reasons for this decrease were explained earlier in Section 4.3.2. For the rest of the map, the radiation intensity decreases according to the inverse square law and is also affected by the material attenuation of air.

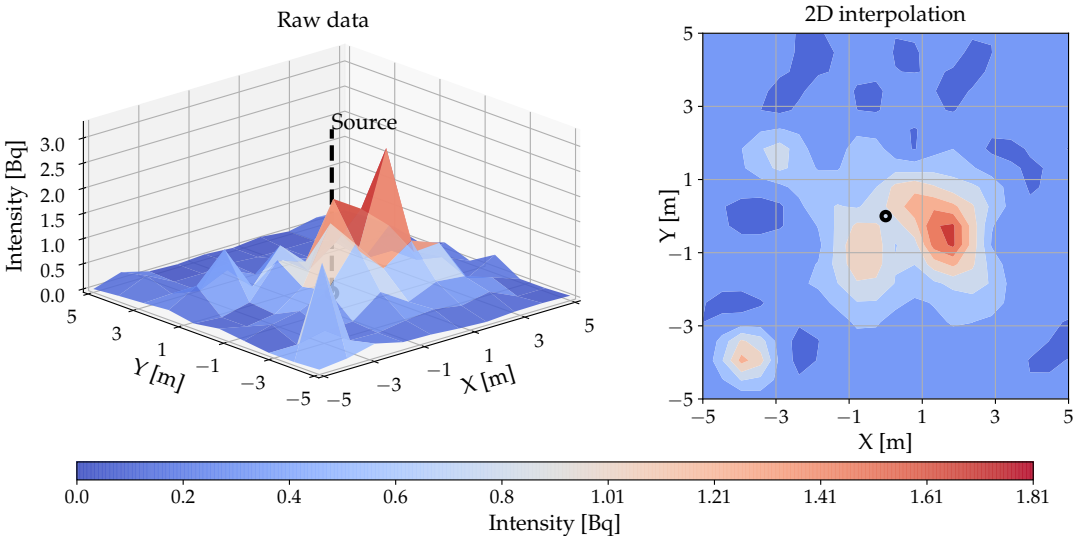


Figure 26 Radiation map created from the measurements obtained during the experiment #2 from the height of 1.5 m. Actual position of the source is highlighted by a black marker. Note the decrease in intensity detected directly above the radiation source.

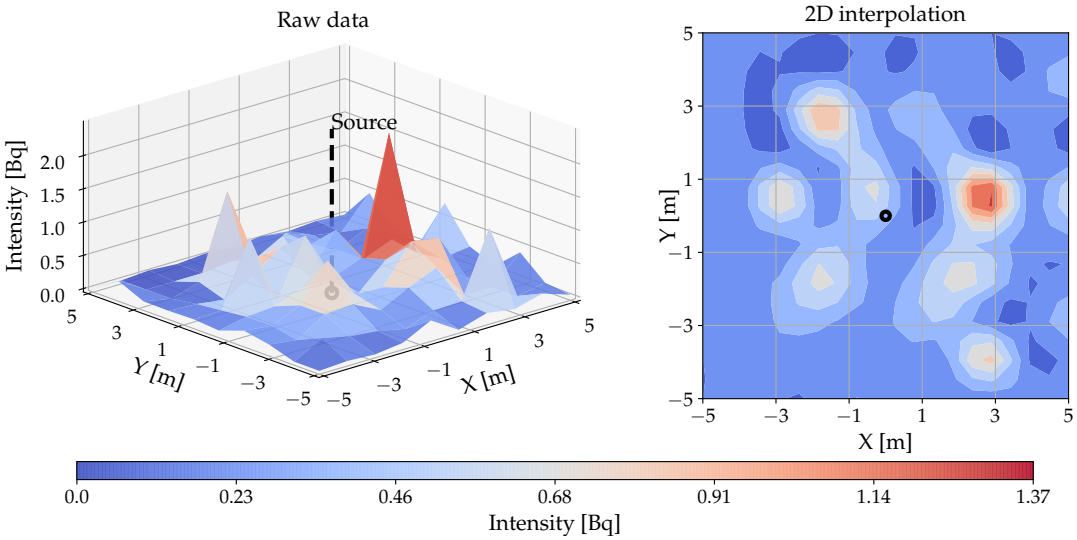


Figure 27 Radiation map created from the measurements obtained during the experiment #3 from the height of 3.0 m. Actual position of the source is highlighted by a black marker. During this experiment, the UAV was moving at a height approaching the technical limit of the optic flow system. At this height, the gamma radiation is also nearly indistinguishable from the natural radiation background.

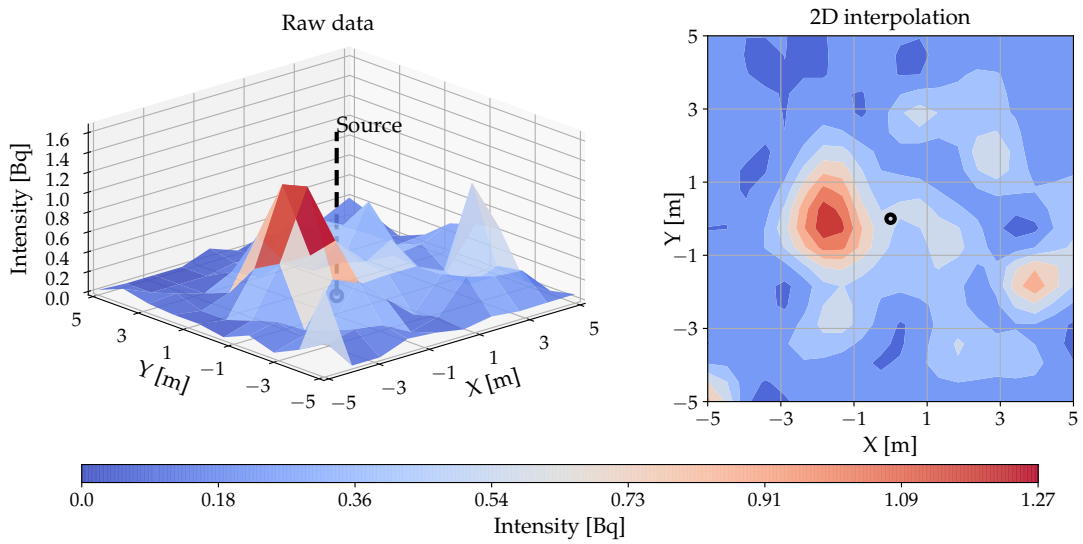


Figure 28 Radiation map created from the measurements obtained during the experiment #4. Actual position of the source is highlighted by a black marker. Despite being performed at the lowest height (0.7 m), the measured intensity did not exceed the measurements from other experiments. The detector was exposed to a direct sunlight during the entire time of the experiment, and a large portion of the pixels was saturated to a maximal value, thereby unable to detect any radiation produced by the ^{241}Am source.

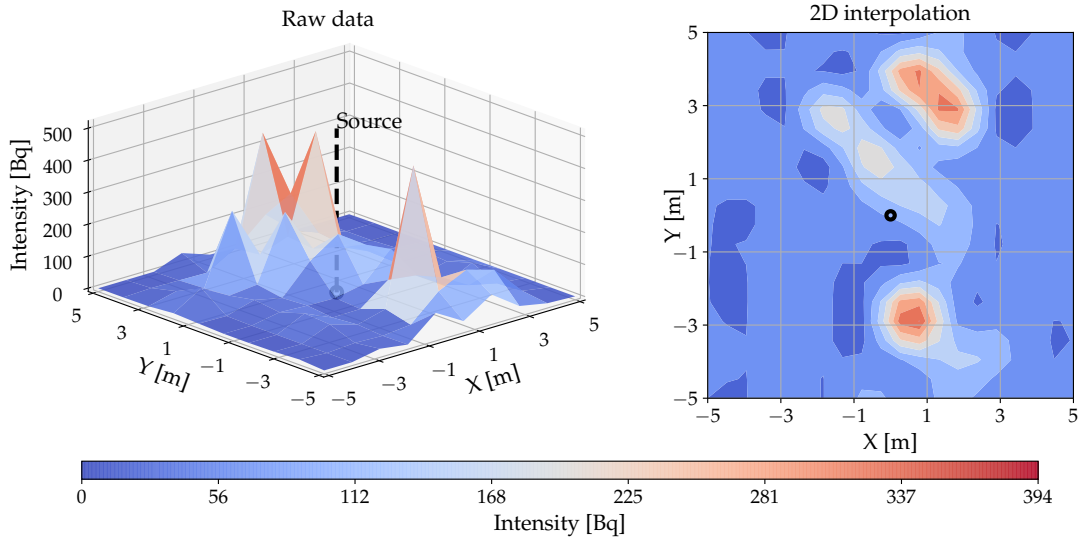


Figure 29 Radiation map created from the measurements obtained during the experiment #5. Actual position of the source is highlighted by a black marker. This experiment was aimed to determine, whether changing the orientation of the sensor will have an effect on the mapping procedure. However, due to the orientation changes, the detector was alternating between exposure to a direct sunlight and a solar shade. These changes produced extreme errors in the measurement, rendering the data unusable. For future missions, a cover from solar radiation is a necessity.

5.1.3 Conclusion

The experiments have proven, that one UAV with a single Timepix detector is sufficient to create a rough map indicating the presence of a radiation source, even when the source

is relatively weak. During the last two flights, the surface of the detector was exposed to a full sunlight, which significantly affected the results. The sunlight causes many of the pixels to reach full saturation, which results in a much smaller detection surface. An example of a Timepix image taken during the experiments is shown in Figure 32.

For this reason, the intensity measured at the lowest height did not exceed the measurements from other flights, despite the detector being the closest to the source. For further experiments, the detector is required to be covered with a protective layer, such as a Kapton, which is commonly used for solar insulation in the space industry. A thin layer would block most of the low energy photons originating from the Sun, and let only the higher energy gamma photons through.

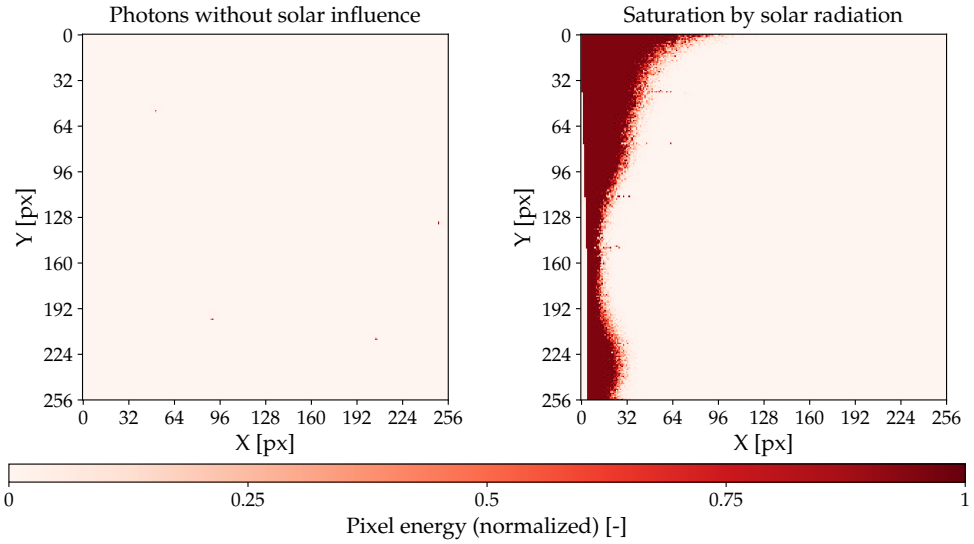


Figure 30 Examples of full Timepix images produced during the experiments. Energy of the pixels is normalized to interval <0,1>. The ideal case is shown on the left. In this image, only photon tracks are detected. The photons leave tracks of one or two pixels in size. On the right, a large area of the detector has been saturated by direct sunlight. Not only does this reduce available detection area, but also results in false track identification along the border of the area.

Figure 31 An outdoor area where the real experiments were conducted. The UAV is highlighted by a red circle. The concrete surface is a square of 10 m × 10 m. The radiation source was placed in the center of the area marked by a green circle.

Figure 32 DJI FlameWheel F450 of the Multi-Robot Systems group. This quadrotor was used to carry the Timepix detector during the real experiments.

5.2 Simulated outdoor environment

The outdoor mapping scenario was recreated in Gazebo using the radiation plugins, and the mapping node running in real-time. The goal was to verify, whether the simulated environment exhibits realistic properties, and can be used for testing of more complex applications.

The parameters of the simulation, including the Timepix exposure time, source material and activity and UAV height and velocity, are set to match the values used in the real experiments. Two results, showing the intensity map acquired from an altitude of 1.5 meters and 3.0 meters are presented in Figure 33 and Figure 34 respectively.

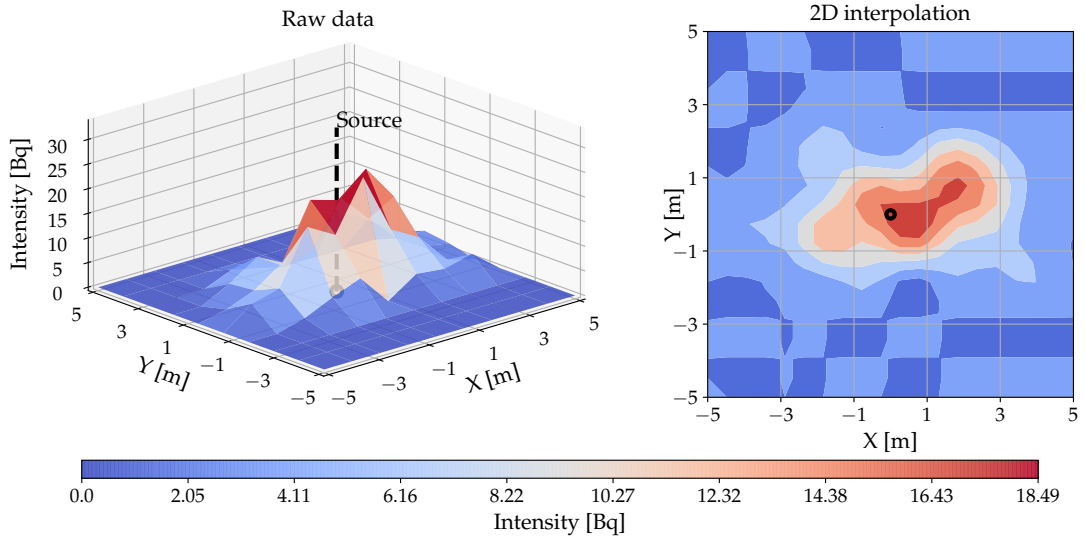


Figure 33 A map of radiation intensity created in a simulated environment with parameters matching the real experiments. During this simulation, the UAV moved at a height of 1.5 m.

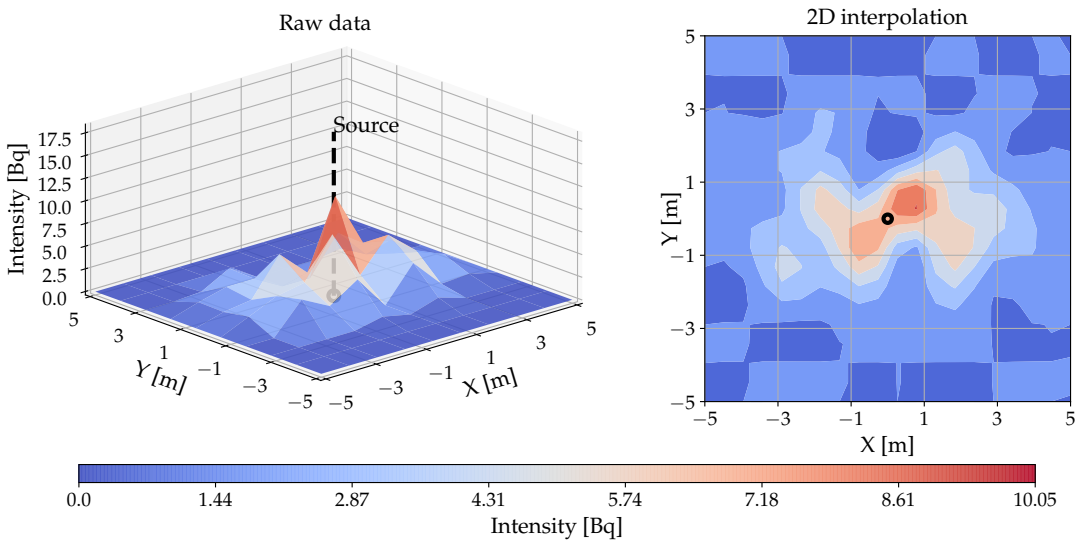


Figure 34 A map of radiation intensity created in a simulated environment with parameters matching the real experiments. During this simulation, the UAV moved at a height of 3.0 m.

5.2.1 Conclusion

The overall behavior of the system closely resembles its real world counterpart. The biggest difference is apparent in the average value of measured radiation intensity. The simulation does not consider radiation background, solar interference or tracks not being recognized by the image classifier. For these reasons, the simulated Timepix detects more particles than real sensor under the same conditions.

5.3 Mapping in indoor environment

This section demonstrates the functionality of the algorithms for mapping and active localization described in Chapter 4. The localization is performed in a simulated environment with obstacles of various properties. The multi-UAV capabilities are demonstrated by deploying a team of two UAVs to solve the task cooperatively. The Timepix is simulated as a 300 µm thick Silicon block, unless stated otherwise.

5.3.1 Environment map

As was mentioned earlier, the map of the scene is known in advance. The UAVs are provided with an occupancy grid of the scene with obstacles expanded by twice the size of the UAV (including the propellers), i.e., by 1.5 m. By planning the path for the UAVs in a grid, the result contains right-angled turns. The controller of the UAV tends to smooth out the corners, so the expansion of obstacles is necessary in order to avoid collisions.

The simulations were performed in an enclosed area of rectangular shape, with a length of 48 m, width of 24 m long and height of 6 m. The map contains sparsely distributed obstacles, which are "transparent" for the gamma radiation. These are represented by 1 cm thick glass panes and 10 cm thick wooden panes. The occupancy grid, with expanded obstacles and the outer walls is shown in Figure 36. In the image, obstacles are assigned colors to distinguish their physical material (blue – glass, brown – wood).

Exactly one radiation source is placed in the scene in every simulation. The position of the source is not known in advance, however, it has to be reachable by the UAVs.

Figure 35 A floor plan of the indoor environment used in the simulations presented in this section. Blue color represents a 1 cm thick glass pane and brown color represents a 10 cm thick wooden wall. Black color represents outer walls, and white represents free space. The obstacles have been expanded by 1.5 meters in all directions to allow a room for deviations in the UAV movement, therefore the difference in thickness is not apparent in the image. The inner dimensions of the area without the obstacle expansion are [24, 48, 6] m.

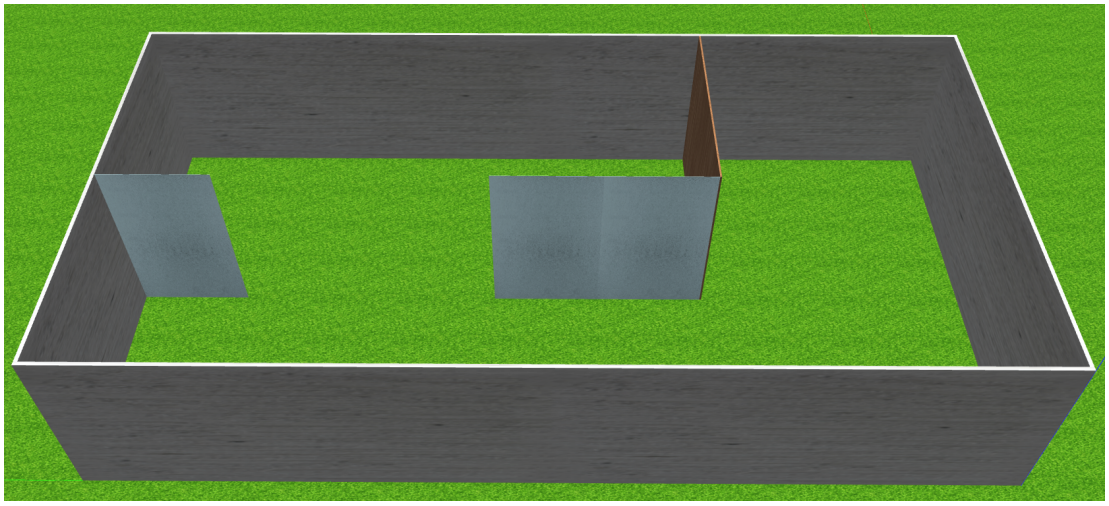


Figure 36 The layout of the scene shown in the GUI of Gazebo. The difference in materials is highlighted by using different textures for the models. The obstacles are shown in their real proportions without expansion.

5.3.2 Cooperative mapping

Just as with the real experiments, the map is divided into grid cells of 1 m^2 in size. A mapping path, shown in Figure 37, was prepared in advance. The UAVs do not change their heading, and the Timepix detector is pointed in the direction of Y axis.

A radiation source with a specific activity of 2 GBq was placed in world position $[17, 19.5, 0]$ m. The simulation was done for both radioisotopes as the source. The map of radiation emitted by ^{241}Am is shown in Figure 38. A map shown in Figure 39 was produced by running the simulation again, but changing the source material to ^{137}Cs.

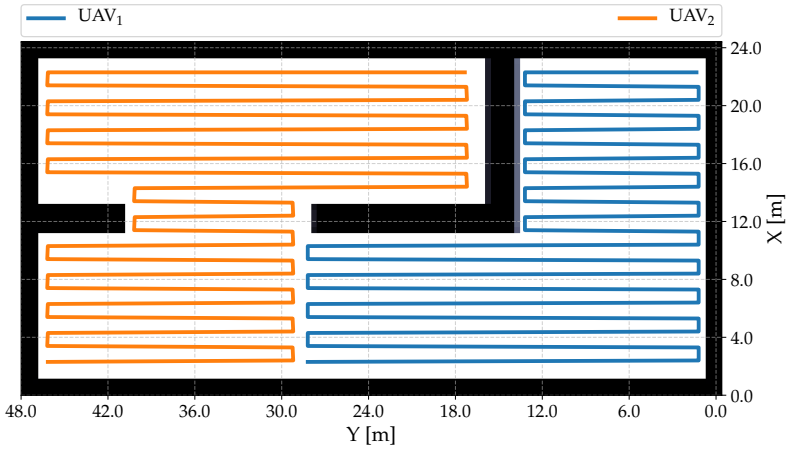


Figure 37 A mapping flight path for two cooperating UAVs. The individual measurements are collected by a central node and combined into one large map.

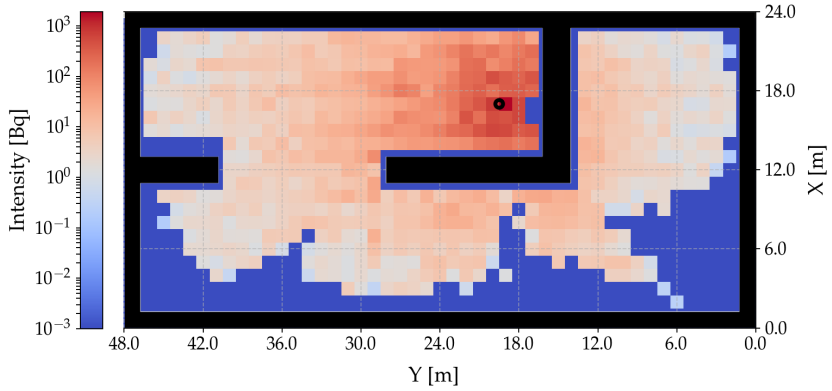


Figure 38 Raw radiation map created by cooperative mapping by 2 UAVs for a simulated ^{241}Am source with activity 2 GBq. The actual position of the source is marked by a black circle. The map clearly shows the intensity attenuation by obstacles, as well as by the orientation of the sensor relative to the source.

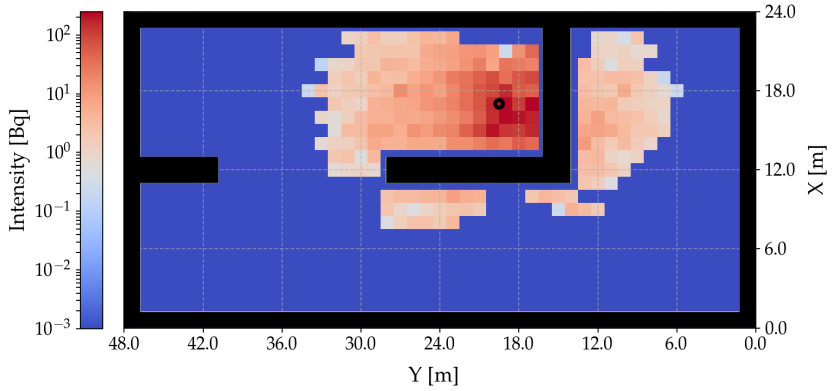


Figure 39 Raw radiation map created by cooperative mapping by 2 UAVs for a simulated ^{137}Cs source with activity 2 GBq. The actual position of the source is marked by a black circle. Although the source produced the same amount of gamma rays as the ^{241}Am in previous simulation, the measured intensity is much lower. This is caused by much higher energy of emitted photons, which mostly pass through the Silicon chip without absorption.

The high energy of photons emitted by the Cesium isotope (662 keV) makes them very difficult to detect with a Silicon chip. In practice, materials with a higher photoelectric absorption coefficient are used instead of Silicon. These materials are for example Cadmium Telluride (CdTe) or Gallium Arsenide (GaAs). The commercially available Timepix variants for high energy imaging use CdTe. Therefore, this material was also used in a simulation. The map created with the use of this sensor is shown in Figure 40. The simulated radiation source is once again ^{137}Cs with activity of 2 GBq. For a final presented result, the simulation was repeated with the same parameters, but the source was placed in position $[11, 32, 0]$ m. The resulting map is shown in Figure 41.

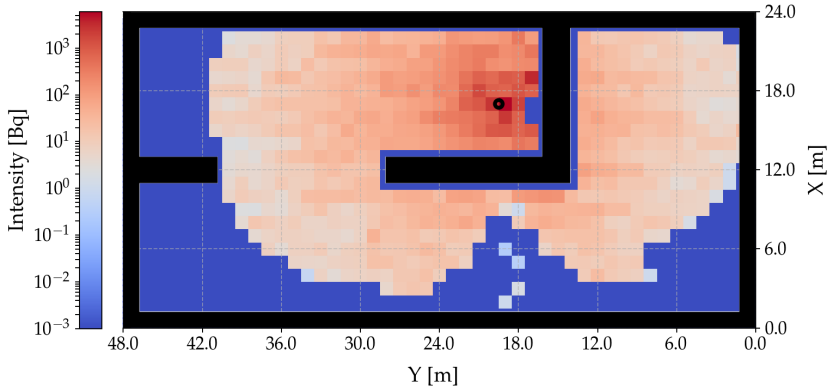


Figure 40 Raw radiation map created by cooperative mapping by 2 UAVs for a simulated ^{137}Cs source with activity 2 GBq. The actual position of the source is marked by a black circle. In this flight, the parameters of the Timepix model were changed. The simulated material of the sensor is Cadmium Telluride (CdTe) with thickness of 1 mm. These changes significantly increase the chance of photoelectric absorption, even for the high energy radiation produced by ^{137}Cs.

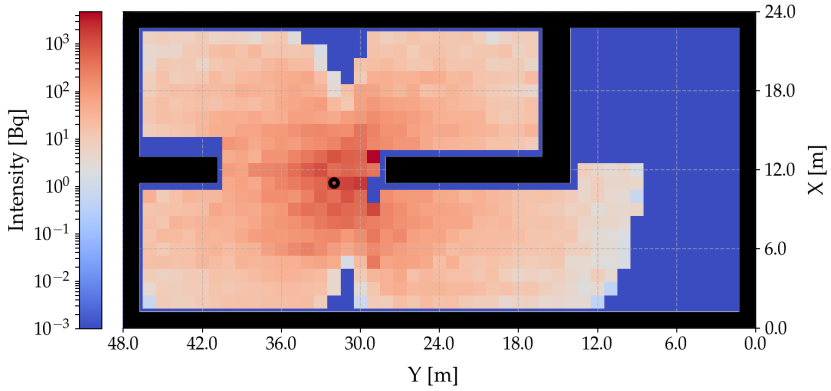


Figure 41 The last experiment repeated with the CdTe sensor. The source was placed in position $[11, 32, 0]$ m. The activity of the source remains at 2 GBq, and the isotope is ^{137}Cs.

5.3.3 Conclusion

This section demonstrates the multi-robot mapping capabilities, as well as the interaction of the simulated radiation and the environment. The attenuation of radiation intensity by the obstacles is apparent from the presented maps. It also demonstrates, that the detection capabilities can be improved by using a sensor of a different material, which is better suited for a specific energy range.

5.4 Active search in indoor environment

In this section, the active searching approach is tested in a simulated indoor environment with obstacles. The simulations were performed in map, which was introduced in the previous section. For each simulation, the source was placed at a randomly selected position on the floor of the world. The material of the source was also selected at random, and the specific activity was set to 1 GBq. Since the source was weaker than the one used in a previous section, the Timepix material was set to 1 mm of CdTe

for all evaluation flights. The increased absorption chance allowed the radiation to be detected at greater distances.

The position estimate from every iteration of the Kalman filter was logged. An estimation error is calculated as an Euclidean distance between the actual position of the source and the position estimate. Evolution of the estimation error during individual simulations is shown in Figure 42. The mean average and the standard deviation of the combined results is also shown in the image.

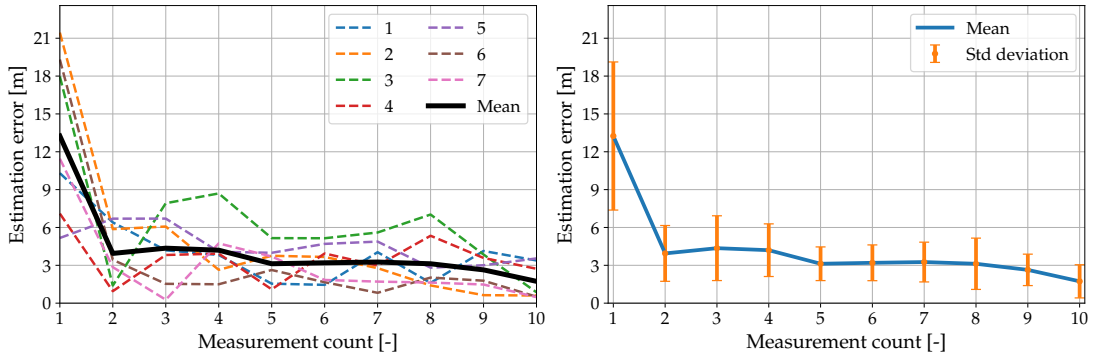


Figure 42 The evolution of an estimation error plotted against the number of performed measurements. The error is calculated as the Euclidean distance between the position estimated by a Kalman filter and the actual position of the source. The estimation error tends to improve with additional measurements.

The process covariance $\mathbf{Q}$, the measurement covariance $\mathbf{R}$, the initial covariance $\mathbf{P}_0$ and the initial state $\mathbf{x}_0$ of the Kalman filter were assigned the following values:

$$\mathbf{Q} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}, \mathbf{R} = \begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 10000 \end{bmatrix}, \mathbf{P}_0 = \begin{bmatrix} 10000 & 0 & 0 \\ 0 & 10000 & 0 \\ 0 & 0 & 10000 \end{bmatrix}, \mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

The iterative improvements of the localization process are shown as an image sequence in Figure 43. The estimated position of the source is improved with each new measurement. The images show orange and blue ellipsoids representing the direction estimates provided by two UAVs (red and blue arrow). The estimated position is shown as a green marker and a purple ellipsoid represents the covariance of the position estimate. The ground-truth of the source is highlighted by a dark blue marker.

5.4.1 Conclusion

The simulations have demonstrated, that the presented solution is capable of estimating the position of the radiation source in real time, with the use of onboard sensors and the algorithms presented in Section 4.3. The estimation error improves with the number of available measurements, therefore it can be concluded, that the estimator works as intended.

The localization process has many phases and the best way to visualize it is in a video. Therefore, a recording of one of the simulations is available online[3] and also on the attached DVD.

[3] `http://mrs.felk.cvut.cz/theses/stibinger2019`

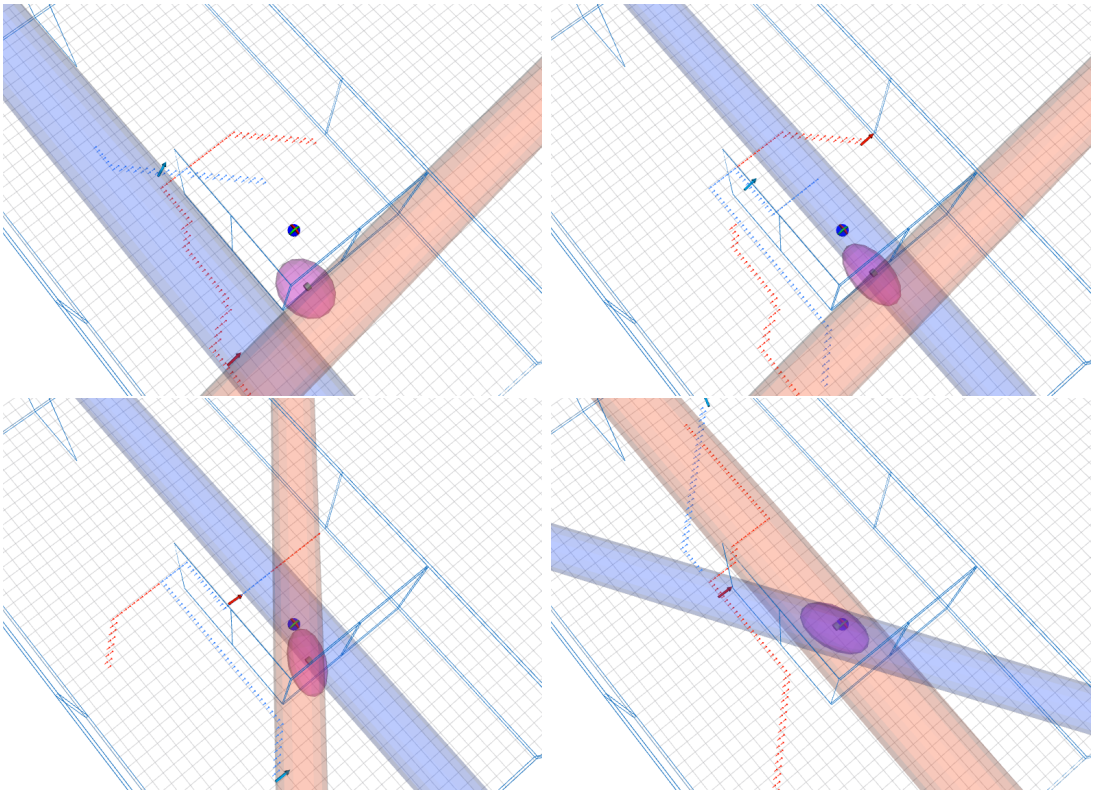


Figure 43 A sequence of images from RVIZ showing the iterative improvement of the estimation with every new measurement. The UAVs are shown as large red and blue arrows. Smaller arrows represent the flight paths between two measurement points. The orange and blue ellipsoids represent the latest direction estimate by the corresponding UAV. The true position of the radiation source is represented by a dark blue marker, while the green marker represents the position estimated by the system, with the purple ellipsoid representing the covariance of the estimate.

6 Conclusion

The goal of this thesis was to develop an autonomous system capable of localizing a source of ionizing radiation. The localization was performed by a group of Unmanned Aerial Vehicles equipped with the hybrid pixel detector Timepix. The thesis aimed to improve the model of the Timepix sensor for the Gazebo simulator.

The new version of the model presented in this thesis implements a raytracing algorithm, which simulates the radiation at the level of individual photons. The implementation allows for simulation of complex environments, where the radiation-obstacle interactions are determined by photon energy and material absorption properties. To complement the detector model, two additional plugins for Gazebo were created – radiation source and radiation obstacle.

The source plugin allows the user to simulate two common radioisotopes – Americium-241 and Cesium-137. The implementation also supports use of multiple different sources at the same time. The obstacle plugin allows the user to add obstacles into the simulated world. Besides the objects provided with the plugin, the user can also select a material from a provided library (includes common materials such as wood, glass, concrete...) and create new obstacles through an automated script. The plugins are released to the public as open-source. The source code is available at a GitHub repository[1].

Properties of the simulated environment were verified by performing real experiments in collaboration with the Czech National Institute for Nuclear, Chemical and Biological Protection. In the experiments, a single UAV equipped with a Timepix detector with a 300 µm Silicon chip followed a predefined trajectory, and measured gamma radiation emitted by a 500 MBq sample of Americium-241. A video from the experiment is available online[2] and on the attached DVD. A dataset collected during one of the experiments is also provided in the form of a rosbag at the website.

The conditions of the experiments were recreated in the simulated environment of Gazebo, and the results of the simulations were compared with the real measurements. Moreover, two control algorithms for a group of UAVs were also developed. The first algorithm is implemented as a map-building ROS node, which collects odometry and Timepix measurements from individual UAVs, and composes them into a shared map of radiation intensity. During this task, the UAVs follow a predefined trajectory. The position of the source can be estimated from the map by finding a global maximum of measured radiation intensity. This approach becomes very time-demanding in larger areas. Therefore a second approach was designed.

This approach utilizes agile movement of the UAVs to estimate the direction of incoming radiation, and implements a linear Kalman filter to fuse the direction estimates to provide an estimated position of the radiation source.

Both localization approaches are not dependent on a specific number of used UAVs, and can be used even with a single helicopter. The algorithms have been thoroughly tested in the simulated environment. Presented results demonstrate, that both algorithms work as intended, and that UAVs equipped with the Timepix detector can indeed

[1] `https://github.com/rospix`

[2] `http://mrs.felk.cvut.cz/theses/stibinger2019`

be used for this type of assignment. Videos from simulations, demonstrating the system in action, are available online[2] and also on the attached DVD.

6.1 Future work

The current system is ready to be tested in a real world scenario. However, it relies heavily on a precise onboard positioning system. Reliable self-localization of the UAV in an indoor environment is currently the most important part of development. In the future, the system may be deployed for inspection of nuclear waste storage facilities, abandoned uranium ore mines or damaged nuclear power plants, such as the Fukushima-Daiichi.

Changing orientation of the sensor relative to the radiation source has proven to work well for estimating the direction of incoming radiation. However, the UAV is unable to maintain a desired pitch and roll in order to estimate the vertical component of the direction. Therefore, the presented localization approach is unable to determine the Z coordinate of the radiation source. Moving the UAVs in a vertical direction would not provide sufficient orientation change, unless the height would change by tens of meters. This approach would not be feasible in an indoor environment, where the system is intended to be used.

To improve the localization process in the vertical direction, the UAV would either need to perform aggressive maneuvers, or to have the ability to independently tilt the Timepix. A different option would be to employ a more advanced sensor, consisting of more Timepix detectors in a stack [56, 57]. Such device is capable of estimating the direction of incoming particles directly, however the heat generated by the sensor requires a cooling mechanism, which would increase the weight of the UAV payload.

[2] `http://mrs.felk.cvut.cz/theses/stibinger2019`

Bibliography

[1] Graeme Stephens et al. “The Department of Energy’s Atmospheric Radiation Measurement (ARM) Unmanned Aerospace Vehicle (UAV) Program”. In: *Bulletin of The American Meteorological Society - BULL AMER METEOROL SOC* 81 (Dec. 2000), pp. 2915–2938.

[2] K Kurvinen et al. “Design of a radiation surveillance unit for an unmanned aerial vehicle”. In: *Journal of environmental radioactivity* 81.1 (2005), pp. 1–10.

[3] Roy Pöllänen et al. “Radiation surveillance using an unmanned aerial vehicle”. In: *Applied radiation and isotopes* 67.2 (2009), pp. 340–344.

[4] Roy Pöllänen et al. “Performance of an air sampler and a gamma-ray detector in a small unmanned aerial vehicle”. In: *Journal of radioanalytical and nuclear chemistry* 282.2 (2009), p. 433.

[5] Peter G Martin et al. “3D unmanned aerial vehicle radiation mapping for assessing contaminant distribution and mobility”. In: *International Journal of Applied Earth Observation and Geoinformation* 52 (2016), pp. 12–19.

[6] Yukihisa Sanada and Tatsuo Torii. “Aerial radiation monitoring around the Fukushima Dai-ichi nuclear power plant using an unmanned helicopter”. In: *Journal of environmental radioactivity* 139 (2015), pp. 294–299.

[7] JW MacFarlane et al. “Lightweight aerial vehicles for monitoring, assessment and mapping of radiation anomalies”. In: *Journal of environmental radioactivity* 136 (2014), pp. 127–130.

[8] Jianyong Jiang et al. “A prototype of aerial radiation monitoring system using an unmanned helicopter mounting a GAGG scintillator Compton camera”. In: *Journal of Nuclear Science and Technology* 53.7 (2016), pp. 1067–1075.

[9] Gordon Christie et al. “Radiation search operations using scene understanding with autonomous UAV and UGV”. In: *Journal of Field Robotics* 34.8 (2017), pp. 1450–1468.

[10] Jinlu Han et al. “Low-cost multi-UAV technologies for contour mapping of nuclear radiation field”. In: *Journal of Intelligent & Robotic Systems* 70.1-4 (2013), pp. 401–410.

[11] Matouš Vrba. “Active Searching of RFID Chips by a Group of Relatively Stabilized Helicopters”. Bachelor’s thesis. FEE CTU in Prague, 2016.

[12] Václav Pritzl et al. “Real-Time Localization of Transmission Sources by a Formation of Helicopters Equipped with a Rotating Directional Antenna”. In: *International Conference on Modelling and Simulation for Autonomous Systesm.* Springer. 2018, pp. 335–350.

[13] Farshad Koohifar, Abhaykumar Kumbhar, and Ismail Guvenc. “Receding horizon Multi-UAV cooperative tracking of moving RF source”. In: *IEEE Communications Letters* 21.6 (2017), pp. 1433–1436.

[14] Zhangjie Fu et al. “Pollution Source Localization Based on Multi-UAV Cooperative Communication”. In: *IEEE Access* 7 (2019), pp. 29304–29312.

[15] Jerry Towler, Bryan Krawiec, and Kevin Kochersberger. “Radiation mapping in post-disaster environments using an autonomous helicopter”. In: *Remote Sensing* 4.7 (2012), pp. 1995–2015.

[16] Martin Saska et al. “Navigation, localization and stabilization of formations of unmanned aerial and ground vehicles”. In: *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2013, pp. 831–840.

[17] Tomas Baca, Giuseppe Loianno, and Martin Saska. “Embedded model predictive control of unmanned micro aerial vehicles”. In: *2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR)*. IEEE. 2016, pp. 992–997.

[18] M. Saska et al. “Swarm Distribution and Deployment for Cooperative Surveillance by Micro-Aerial Vehicles”. In: *Journal of Intelligent & Robotic Systems.* 84.1 (2016), pp. 469–492.

[19] V. Spurny et al. “Cooperative Autonomous Search, Grasping and Delivering in a Treasure Hunt Scenario by a Team of UAVs”. In: *Accepted in Journal of Field Robotics* (2018).

[20] Stephen M Seltzer. *X-Ray Mass Attenuation Coefficients NIST Standard Reference Database 126*. 2004. URL: `https://www.nist.gov/pml/x-ray-mass-attenuation-coefficients` (visited on 12/03/2019).

[21] W Stempfhuber and M Buchholz. “A precise, low-cost RTK GNSS system for UAV applications”. In: *Proc. of Unmanned Aerial Vehicle in Geomatics, ISPRS* (2011).

[22] Marvelmind Robotics. *Precise Indoor GPS for autonomous robots, drones and VR*. 2017. URL: `https://marvelmind.com/` (visited on 03/04/2019).

[23] M. Saska et al. “System for deployment of groups of unmanned micro aerial vehicles in GPS-denied environments using onboard visual relative localization”. In: *Autonomous Robots* 41.4 (2017), pp. 919–944.

[24] Matěj Petrlík. “Onboard Localization of an Unmanned Aerial Vehicle in an Unknown Environment”. Master’s thesis. FEE CTU in Prague, 2018.

[25] Vikrant More et al. “Visual odometry using optic flow for unmanned aerial vehicles”. In: *2015 International Conference on Cognitive Computing and Information Processing (CCIP)*. IEEE. 2015, pp. 1–6.

[26] Glenn F Knoll. *Radiation Detection and Measurement; 3rd ed.* New York, NY: Wiley, 2000. URL: `https://cds.cern.ch/record/441925`.

[27] Fred A Mettler Jr et al. “Radiologic and nuclear medicine studies in the United States and worldwide: frequency, radiation dose, and comparison with other radiation sources—1950–2007”. In: *Radiology* 253.2 (2009), pp. 520–531.

[28] T Poikela et al. “Timepix3: a 65K channel hybrid pixel readout chip with simultaneous ToA/ToT and sparse readout”. In: *Journal of instrumentation* 9.05 (2014), p. C05013.

[29] Zdenek Vykydal, Jan Jakubek, and Stanislav Pospisil. "USB interface for Medipix2 pixel device enabling energy and position-sensitive detection of heavy charged particles". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 563.1 (2006), pp. 112–115.

[30] Zdenek Vykydal and Jan Jakubek. "USB Lite—Miniaturized readout interface for Medipix2 detector". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 633 (2011), S48–S49.

[31] T Baca et al. "Miniaturized X-ray telescope for VZLUSAT-1 nanosatellite with Timepix detector". In: *Journal of Instrumentation* 11.10 (2016), p. C10007.

[32] Martin Urban et al. "VZLUSAT-1: Nanosatellite with miniature lobster eye X-ray telescope and qualification of the radiation shielding composite for space application". In: *Acta Astronautica* 140 (2017), pp. 96–104.

[33] Rafael Ballabriga, Michael Campbell, and Xavier Llopart. "Asic developments for radiation imaging applications: The medipix and timepix family". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 878 (2018), pp. 10–23.

[34] Nicholas Stoffle et al. "Timepix-based radiation environment monitor measurements aboard the International Space Station". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 782 (2015), pp. 143–148.

[35] J Zemlicka et al. "Analysis of painted arts by energy sensitive radiographic techniques with the Pixel Detector Timepix". In: *Journal of Instrumentation* 6.01 (2011), p. C01066.

[36] Srinidhi Bheesette et al. "Medipix3RX neutron camera for ambient radiation measurements". In: Oct. 2017, pp. 1–5. DOI: 10.1109/NSSMIC.2017.8533085.

[37] T Baca et al. "Rospix: modular software tool for automated data acquisitions of Timepix detectors on Robot Operating System". In: *Journal of Instrumentation* 13.11 (2018), p. C11008.

[38] Martin Jílek. "Processing of Radiation Data from the Timepix Sensor on the VZLUSAT-1 Satellite". Master's thesis. FEE CTU in Prague, 2018.

[39] TullyFoote. *Robot Operating System documentation*. 2018. URL: http://wiki.ros.org/ (visited on 04/10/2019).

[40] Inc. The MathWorks. *Robot Operating System in Matlab*. 2019. URL: https://www.mathworks.com/help/robotics/robot-operating-system-ros.html (visited on 04/10/2019).

[41] Open Source Robotics Foundation. *Gazebo robot simulation made easy*. 2014. URL: https://gazebosim.org/ (visited on 04/11/2019).

[42] Google. *Protocol Buffers*. 2014. URL: https://developers.google.com/protocol-buffers/ (visited on 04/11/2019).

[43] Indrajeet Yadav et al. "Visual-Inertial Target Tracking and Motion Planning for UAV-based Radiation Detection". In: *arXiv e-prints*, arXiv:1805.09061 (May 2018), arXiv:1805.09061. arXiv: 1805.09061 [cs.RO].

[44] Richard S Quimby. *Photonics and lasers*. Wiley Online Library, 2006.

[45] Petr Štibinger. "Localization of a Radiation Source by a Formation of Unmanned Aerial Vehicles". Bachelor's thesis. FEE CTU in Prague, 2017.

[46] Rudolph Emil Kalman. "A new approach to linear filtering and prediction problems". In: *Journal of basic Engineering* 82.1 (1960), pp. 35–45.

[47] Harold Kushner. "Approximations to optimal nonlinear filters". In: *IEEE Transactions on Automatic Control* 12.5 (1967), pp. 546–556.

[48] Greg Welch, Gary Bishop, et al. "An introduction to the Kalman filter". In: (1995).

[49] Simon J Julier and Jeffrey K Uhlmann. "New extension of the Kalman filter to nonlinear systems". In: *Signal processing, sensor fusion, and target recognition VI*. Vol. 3068. International Society for Optics and Photonics. 1997, pp. 182–194.

[50] Eric A Wan and Rudolph Van Der Merwe. "The unscented Kalman filter for nonlinear estimation". In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. Ieee. 2000, pp. 153–158.

[51] Branko Ristic, Sanjeev Arulampalam, and Neil Gordon. "Beyond the Kalman filter". In: *IEEE Aerospace and Electronic Systems Magazine* 19.7 (2004), pp. 37–38.

[52] Taeyoung Lee. "Geometric tracking control of the attitude dynamics of a rigid body on SO (3)". In: *Proceedings of the 2011 American Control Conference*. IEEE. 2011, pp. 1200–1205.

[53] T Baca et al. "Model Predictive Trajectory Tracking and Collision Avoidance for Reliable Outdoor Deployment of Unmanned Aerial Vehicles". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–8.

[54] Tomas Baca et al. "Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 6753–6760.

[55] Vojtěch Spurný et al. "Cooperative autonomous search, grasping, and delivering in a treasure hunt scenario by a team of unmanned aerial vehicles". In: *Journal of Field Robotics* 36.1 (2019), pp. 125–148.

[56] P Soukup, J Jakubek, and Z Vykydal. "3D sensitive voxel detector of ionizing radiation based on Timepix device". In: *Journal of Instrumentation* 6.01 (2011), p. C01060.

[57] D Turecek et al. "Compton camera based on Timepix3 technology". In: *Journal of Instrumentation* 13.11 (2018), p. C11022.