SAVONIA

University of Applied Sciences

THESIS – BACHELOR'S DEGREE PROGRAMME
TECHNOLOGY, COMMUNICATION AND TRANSPORT

SIMULTANEOUS LOCALIZATION AND MAPPING TECHNIQUES WITH AI-BASED STEREO CAMERA

AUTHOR Timon Padberg

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology, Internet of Things			
Author(s) Timon Padberg			
Title of Thesis Simultaneous Localization and Mapping Techniques with AI-Based Stereo Camera			
Date	31 January 2024	Pages/Number of appendices	29
Client Organisation /Partners University of Turku, Savonia University of Applied Sciences			

Abstract

This thesis involved incorporating two distinct stereo camera systems, namely ZED-X and ZED-2i, through the utilization of SLAM technology to create a three-dimensional representation of an environment. The primary goal was to enable the systems to generate a 3D map using point clouds and accurately track the position, specifically for robotics applications. Two experimental setups were created with distinct configurations to facilitate comparison and interpretation of the results.

The utilization of the SLAM technology within the robotic operating software played a pivotal role in augmenting the system's performance. The system was structured by developing dataflow diagrams and establishing a workspace in ROS. Both systems underwent testing in an indoor setting, while the ZED-2i system was additionally tested outdoors.

The experimental findings for this thesis were remarkable, as the ZED-X demonstrated precise visual outcomes. Within the ZED-2i system, there were instances of erroneous identifications and a reduced level of point cloud density compared to the ZED-X. During the testing phase, both setups exhibited favorable and elevated rates of detecting the depth and positioning of objects. Despite the limitations observed in both setups during the experiments, these challenges can be addressed through additional investigation and more comprehensive testing. One of the proposed recommendations for future research was the implementation of the ZED-2i and ZED-X stereo cameras for a robotic system.

Keywords
IoT, AI, Stereo Camera, ZED-2i, ZED-X, SLAM, RTABMAP, Jetson Orin NX, Point cloud, 3D Mapping, Python, C++, ROS, NVIDIA, Linux.

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

In this first chapter the introduction to the research topic is given including the background and motivation of starting the thesis. After the objectives and scope are defined it creates a better understanding of the areas that are considered and taken into account.

1.1 Background and Motivation

There is not much information available about the topic, since the products used in this research are extremely new and recently released. Older versions of the products will be looked at as background information for setting up the system and testing its functionalities. One of the researches that is being looked at, and referred to more often in this thesis, as relevant background information is: "Study on Stereo AI-Based Zed-2i Camera". This study analyzes how to create the environment and the implementation of SLAM with ZED-2i camera as part of this thesis. This article will be published around the same time as the thesis is finalized. Knowledge gained during this research is used as a base for this thesis. The software ROS stands for Robotics Operating System and has many functionalities and ways of communicating with robots. There are software packages available that are open-source for ROS that are commonly used by the community and the supplier recommends some packages. This framework and documentation will be used as the software is familiar and suitable for this research.

1.2 Research Objectives

In this research, the aim is to get detailed results that realistically show the environment into a map with point clouds and correct positional tracking suitable for robotics. The main objective is to get the system to generate a 3D map by implementing SLAM technology with the ZED-2i and ZED-X stereo cameras. The secondary objective is to compare the results of the ZED-2i and the ZED-X looking at details and accuracy of the environment. The proposed result would be a breakdown of the working of the systems and experimental results. The proposed system is ready to be used and implemented on a robot. An example of a successful result is a database file, recording of the experiment process, or pictures of experiment results.

1.3 Scope and Limitations

The scope of this research is to create test environments for two different systems to be able to analyze and compare the results. The results are analyzed based on observations and details taken from the snapshots. Investigations and in-depth analyzes will be outside of the scope of this research. Challenges will most likely be faced in the system with the microcontroller and the latest technology it uses.

1.4 Literature Survey

In autonomous robots, stereo cameras enhanced by AI technology have the potential to create new ways of detecting such as objects and obstacles. The depth perception from a stereo camera and the analytical power of AI algorithms give robots the ability to represent a better image of the environment and interact to make more precise avoidance maneuvers because of the higher precision in mapping and localization.

In previous research, it shows that RTAB-Map, in combination with stereo cameras in an outdoor environment, accurately measures the location traveled by a robot (de Jesus, Kobs, Cukla, de Souza Leite Cuadros, Gamarra 2021). Another research study also mentioned that RTAB-Map appears to be a fast approach to creating 3D maps, which could be helpful in certain situations (Wolf et. al 2020). The experiment showed a slight offset of the original path while turning on the horizontal axis. This article focuses on singular axis movements of the camera that simulate the use case of a four-wheeled robot. RTAB-Map is superior in translational movements but has a higher error rate for rotational movements (Altuntaş, Uslu, Çakmak, Amasyalı, Yavuz 2017). Characteristics of RTAB-Map regarding the localization are sufficient as demonstrated with different hardware-limited configurations (Ramachandran, Sahin 2019), (Chiodini et. al 2021). Reviewed material displays that the accuracy might vary depending on the camera used for visual mapping in indoor environments (Zhang et. al 2022). RTAB-Map shows an example of positional recognition and path tracking with the ZED 2 stereo camera with the robot's corresponding error rate and response time (Firdaus, Hutagalung, Syahputra, Analia 2023). The older version of the ZED camera has also been analyzed on depth error registration at multiple depth levels (Fernandez, Avila, Goncalves 2017). These results are valuable to be analyzed for this thesis results based on the instabilities of the experimental setup. Other features in the stereo camera are the depth sensing qualities demonstrated on an electric socket (Tadic 2023).

2 TECHNOLOGIES DEFINITION

In this chapter the technologies will be briefly explained to describe and create a foundation of the working process of the software used. These technologies are implemented in the experiments and are executed and actively doing complex computational calculations in the background.

2.1 Simultaneous localization and mapping

The main method for localizing and mapping an area at the same time is the use of SLAM (Simultaneous Localization And Mapping). SLAM is being used for example in autonomous vehicles, robots or drones to map an environment. This technology enables multiple data inputs to be processed and creates a virtual environment that will be recognized through the use of algorithms. It will know certain positions on the map, and based on what is known estimate the robot’s position through odometry in the camera. Besides autonomous robots, SLAM is also applied in self-driving cars, augmented reality and many more cases (Flyability, n.d.). In for example a more common product such as a vacuum cleaning robot can make a huge difference. By using SLAM the robot understands the layout of the room and creates a path in this area based on its surroundings to make it more efficient. It prevents going over the same spot multiple times, saves time and preserves energy. Figure 1 captures the core of the SLAM process as described.

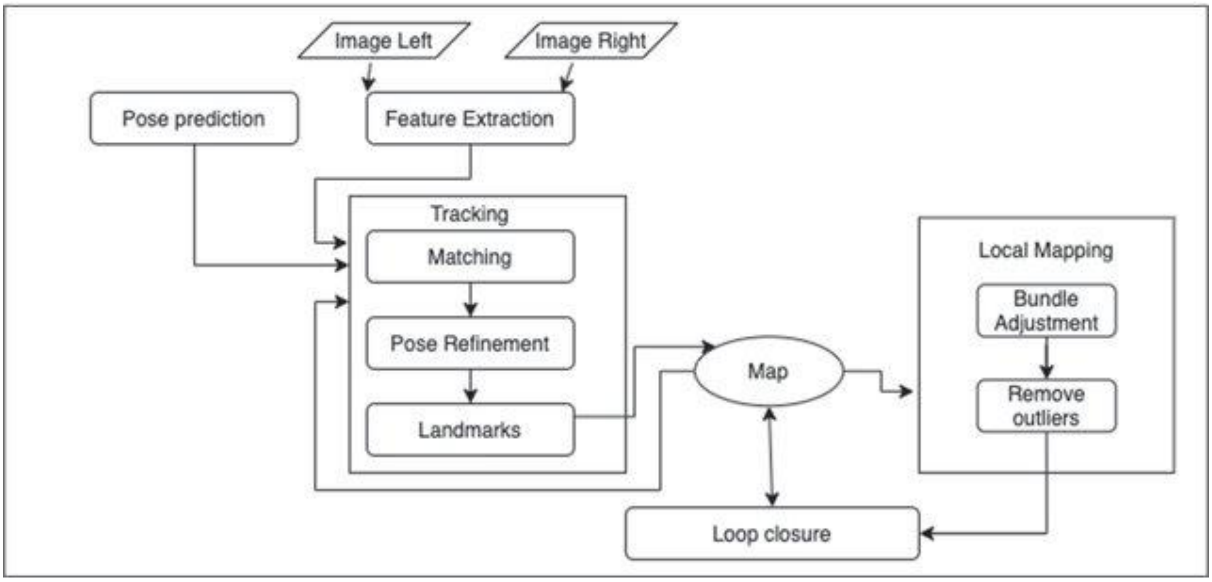

Figure 1. Simplified flowchart of VSLAM (Rohit Yadav 2022)

2.1.1 Localization

A robot using SLAM localizes itself by continuously updating of its position. In this sub-section a more in-depth explanation of the working of the localization will be given. The process of understanding directions uses algorithms and sensor data as input. The sensor data that is being used is to determine the position of the Odometry and the inertial measurement unit (IMU). The IMU is being used to determine the orientation of the camera. Values that the IMU measures are the acceleration in x, y and z and also the rotational acceleration roll, pitch and yaw. Certain cameras and LIDAR devices have this IMU integrated and can be accessed quite easily. If this is not available another option is to get the current position based on the captured information of the surroundings. Variables such as depth and distance of points in the environment are used to get the direction and location the device is facing. There are also other features of collecting landmarks to optimize the localization. Algorithms can identify known landmarks and create a relation between the position of known and unknown points in the environment. Loop closure is a commonly used technique to correct the current position. The position is already scanned and whenever the robot revisits the location it remembers certain points in the environment that it has seen before.

The process of estimating and defining the location will be done iterative while the robot is moving and collecting data. By combining the information from the features, the robot is able to localize itself even without GPS.

2.1.2 Mapping

During the current localization of the robot as described in the previous sub-section the algorithm creates a virtual map of the environment. In this map, the location points of the robot are marked in a set interval as a point. Algorithms can use these location points to draw a line to create a path that the robot has been traveling. As mentioned, the localization is done simultaneously while observing the environment through the precision improvements of the features mentioned. The moving robot updates its map with new observations continuously. The intake of points in a slightly different angle makes map fusion possible. Map fusion unifies information in the map to give a consistent map that reflects the most accurate and up-to-date information. Map fusion also minimizes the effect of movement by updating parts in the map that are in line with the latest input of that specific area. The map can be used in combination with localization to help the robot navigate autonomously. Based on the information in the map the robot understands its surroundings and plans a path to avoid obstacles to reach its destination.

2.2 Point cloud

The map as specified is basically empty and just filled with localization positions points of the robot. Point cloud is a way to generate 3D points of cloud from values extracted from the environment. Each cloud contains coordinate values (x, y, z) representing its place in the map. The distance from the sensor is calculated through the depth recognition of the signal meaning the time it takes for it to return (MathWorks, 2023). The cloud can also contain information about the color and intensity of the signal. Generally speaking, whenever the sensor collects enough clouds with a certain density a cluster is formed. Clustering clouds makes it easier to recreate surfaces and assign clouds to a certain object. These objects can be analyzed and become waypoints, or landmarks, for SLAM to instruct the robot to interact or avoid. Examples of interactions with these objects are object detection, object recognition or augmented reality.

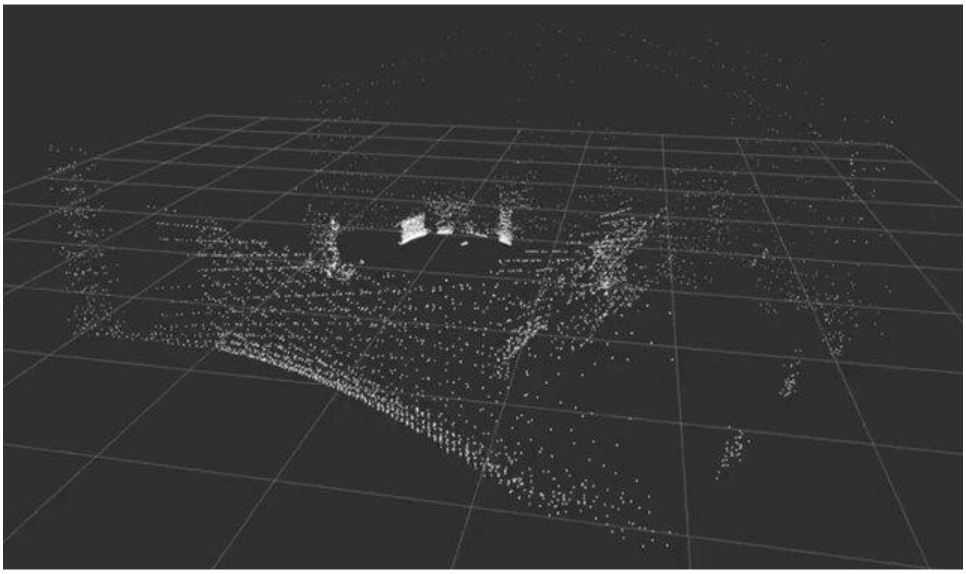

Figure 2. Point cloud visualized in ROS (J. Pena Queralta, 2019)

Figure 2 shows a way of representing collected clouds. Multiple parameters have to be set during the visualization to improve the accuracy of real objects.

2.3 Algorithms

SLAM makes use of algorithms that have a designated task in the system to process incoming data. Odometry, vision and mapping are examples of mathematical algorithms that are executed and operational while running technologies enabling this feature. The framework of SLAM algorithms is often done in C++ and Python programming languages. The reason for using these languages is mainly related to good performance and compatibility of the language. Depending on the implementation another language might be more suitable. C++ codes give the advantage over other languages because of low-level memory accessibility, speed of iterations, real-time processing of data and relatively seen as the main language used in SLAM libraries together with Python. The function of Python language in SLAM makes it easy to deploy and develop algorithms. Python, like C++, has a decent amount of libraries created by the community because of its compatibility with OpenCV. While C++ handles the lower level of the system, Python controls the higher-level code in Figure 1.

2.4 ROS (Robot Operating System)

ROS, which stands for Robot Operating System, is a versatile toolset for developers to interact with robots. ROS is available in different versions and distributions specific to the operation system. According to the previously mentioned OS version running on the system, the suitable ROS distribution would be Noetic. Noetic is compatible with Ubuntu 20.4 and Python 3 and is the latest supported release. Opting for the complete Noetic installation guarantees that the most essential packages are included for 2D and 3D simulations. This paper focuses on 3D simulations, making the entire structure the preferred option to eliminate the need to download fundamental packages individually. Official pages provide installation guides to simplify the installation and setup procedure. After installation of the environment, it needs to be sourced, which is recommended to be automated to manage dependencies. Establish a new workspace, serving as a place to install other packages from external sources.

In order to use ROS nodes, roscore must run for the nodes to be able to communicate with each other. Roscore is the master branch through which different services and data will be exchanged (LucasWalter, 2019). ROS allows users to create packages that contain nodes are easy to reuse and share with others. These packages are installed relatively easily in the ROS workspace by the use of the "catkin_make" command. ROS also provides the possibility to visualize through RVIZ (ROS Visualization). RVIZ enables the user to visualize the state of a robot or device. The visualization can be a recording or real-time example of the environment observed by the robot (Dave Hershberger, 2023). A simulation is the virtual variant of a recording for testing purposes of the robot model. The features that ROS provides are perfect for research, development and testing of robots.

2.4.1 Publishing topics

Publishing topics are a fundamental mechanism in ROS for communicating between nodes. Algorithms and other service nodes that enable features have to communicate to exchange information to be able to extract and use the data. Nodes that are published, generate data to a topic that is receiving the data. One node can subscribe and publish to multiple topics. A tool to get all the published messages to a topic the following command can be used in the terminal "rostopic echo /topic_name" (ROS Topic command line tools – practical example (rostopic and rosmsg), 2023).

2.4.2 Subscribing to topics

As a published node sends information, the subscriber node receives or listens to the information broadcasted by the publisher node. The subscriber node uses the information as an input and uses it in the algorithm it is running. The subscriber node publishes the processed information and sends it to another node. Connecting nodes in this typical way a web or chain of nodes is created, creating useable and processed data as a result.

3 SYSTEM ARCHITECTURE

In this chapter the system's architecture will be described including a breakdown of the data flow and components used in two different systems.

3.1 Architecture of ZED-2i system

The ZED-2i system has a simple and basic structure at the hardware and software level. The flow of the data has only the core parts of the system in it to simplify the model. It starts with the camera sending data to the workstation that runs and processes the data to be used by a robot. A flowchart of the data can be found in Figure 3. The dotted lines in the flowchart indicate when information is being passed and pulled from software to hardware and vice versa. This dotted line is the alternative route of the data while it is being processed. The straight connector line represents the direct flow that the data follows in the designated container.

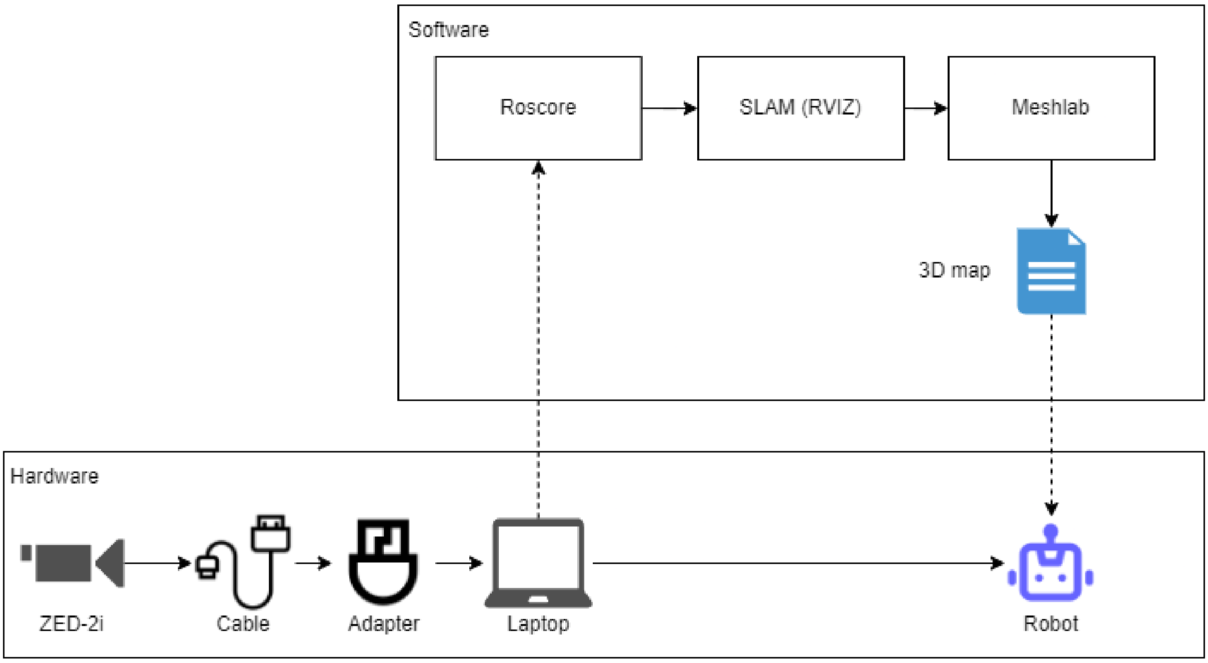

Figure 3. Data flowchart of ZED-2i system

3.1.1 Hardware interface of ZED-2i system

In the ZED-2i system the hardware consists of a laptop that is connected through a USB cable with a stereo camera. The laptop and camera cannot be connected directly because of the missing USB port on the laptop. In order to make the equipment compatible is through the use of an adapter that allows the camera to transfer data to the laptop. The technical specifications of each part can be found in Table 1.

Table 1. Hardware specifications of ZED-2i experiment

Parameters	**Configuration**
Computing system	
Computer model	Dell XPS-15-9510
Processor	11th Gen Intel® Core™ i7-11800H @ 2.30 GHz
GPU	NVIDIA GeForce RTX 3050 TI
RAM	32 GB DDR4
Camera equipment	
Camera	Stereolabs ZED-2i
Connector cable	USB-C 3.0, 1.5 meter locking cable
Adapter	DELL USB-C to USB-A adapter

The ZED-2i stereo camera has a USB3.0 Type-C connection output. The connection cable will be connected on the camera side through USB-C and in the laptop through USB-A. The USB-C section of the cable has two screw locks to prevent the cable from detaching. Transfer speed is ultra-fast with rates up to 5 Gpbs. The USB connector type makes this a versatile option to connect to any robot or microcontroller.

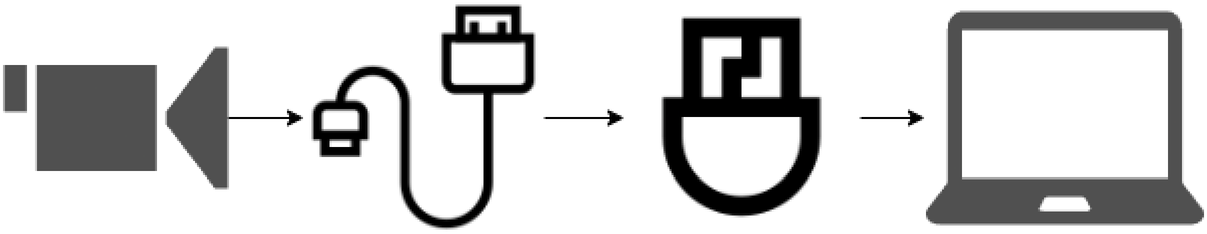

Figure 4. Data flow diagram of ZED-2i architecture

In Figure 4 depicts the hardware part of the data flow and direction through the components. The path of the data is short making minimizing the possibility of lag through the system.

Figure 5. Reconstruction of ZED-2i experiment setup

A visual representation of the setup can be seen in Figure 5. The connection is simple with not a lot of wiring and is modular with a good reach given the length of the cable.

3.1.2 Software-related matters

The choices of operating systems are limited due to the compatibility requirements of other essential software. Linux-based operating systems are compatible and versatile with robotics software. This research utilizes ROS1, Robotics Operating System 1, compatible with Ubuntu versions 20.4 and 24.4. These Ubuntu versions are LTS, Long Term Services, and provide support and essential updates for longer. Obtain the ISO file of the correct Ubuntu version directly from the official source. Utilize an open-source tool for creating a bootable and put this on a removable memory drive. The installation steps should be followed closely, but make sure to allocate sufficient cores and memory to ensure this won't be a limiting factor during the experiments. Furthermore, using the Ubuntu Pro option is recommended, offering additional benefits such as security updates that last multiple years.

3.1.2.1 NVIDIA drivers

Selecting the suitable configuration of NVIDIA drivers is essential to smooth operations. Without the correct driver, the operating system won't recognize the GPU. The available drivers can be accessed via "Software and Updates" under the additional drivers tab. The driver's newest version at installation can be selected, but avoid choosing the open kernel version. Be aware that the NVIDIA driver updates might lead to issues if not addressed in advance.

Blacklist the Nouveau driver by creating a configuration file with the parameter "nouveau modeset=0" as shown in Figure 6.

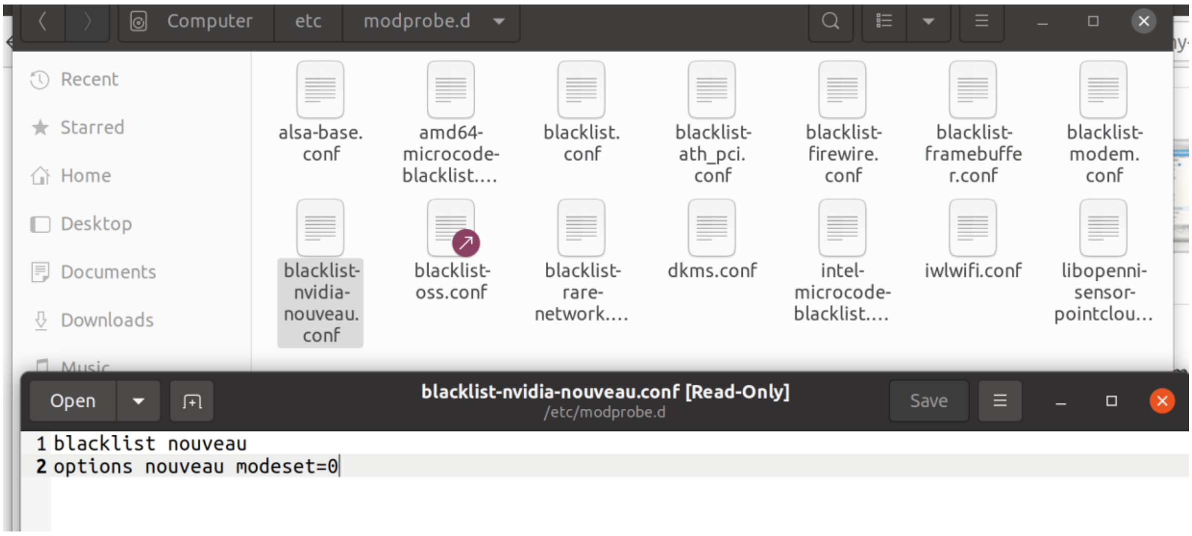

Figure 6. Nouveau driver blacklisting

The next step is to confirm the current version of the driver and if the GPU is detected by using the following command in the terminal "nvidia-smi". If the message "GPU or drivers not detected" appears, we need to see the additional drivers tab and downgrade if necessary. In this research, the driver version 525.125.06 has been used as this is a stable release at the point of testing. In case of sleep mode activation or driver detection problems, restarting the system often solves driver-related problems. If this does not solve the issue, we also took assistance from the community forum to find the proper solution.

3.1.2.2 CUDA toolkit

CUDA, which stands for Compute Unified Device Architecture, toolkit from NVIDIA plays a vital role in facilitating the total computing power of a GPU. The Stereolabs SDK requires the libraries and architecture that CUDA enables. Integrating parallel GPU processing supplies the deep learning algorithms with the computational power needed. Selecting the corresponding CUDA version according to the other software versions mentioned earlier is crucial. Currently, ZED SDK 4.0 supports CUDA version 12.0 and is also compatible with Ubuntu 20.4. Older versions of CUDA that are working on the Ubuntu-18 platform are deprecated, which makes it not recommended to use. The CUDA deep learning models are ideal for the ZED-2i stereo camera, which operates, for example, imaging, color correction, and visualization floating-point computing. Issues regarding the NVIDIA driver also apply to the CUDA installations, with instances in the operating system that do not detect the software. Verifying the detection of CUDA can be done utilizing the terminal in the same way as described with the other drivers.

3.2 Architecture of ZED-X system

The architecture of the second system, also referred to as the ZED-X system, is slightly different than the previous system. This system has a direct connection to the hardware section and multiple steps in the software section of the diagram. The main structure of this system is similar but the path of the data is unique. The ZED-X flowchart is depicted in Figure 7, with a robot acting based on the information given. The dotted lines in the flowchart indicate, as it did in Figure 3, information that flows through and is pulled from software to hardware and vice versa. Also known as the alternative data path to be processed. The straight connectors are used to direct the data flow within the container.

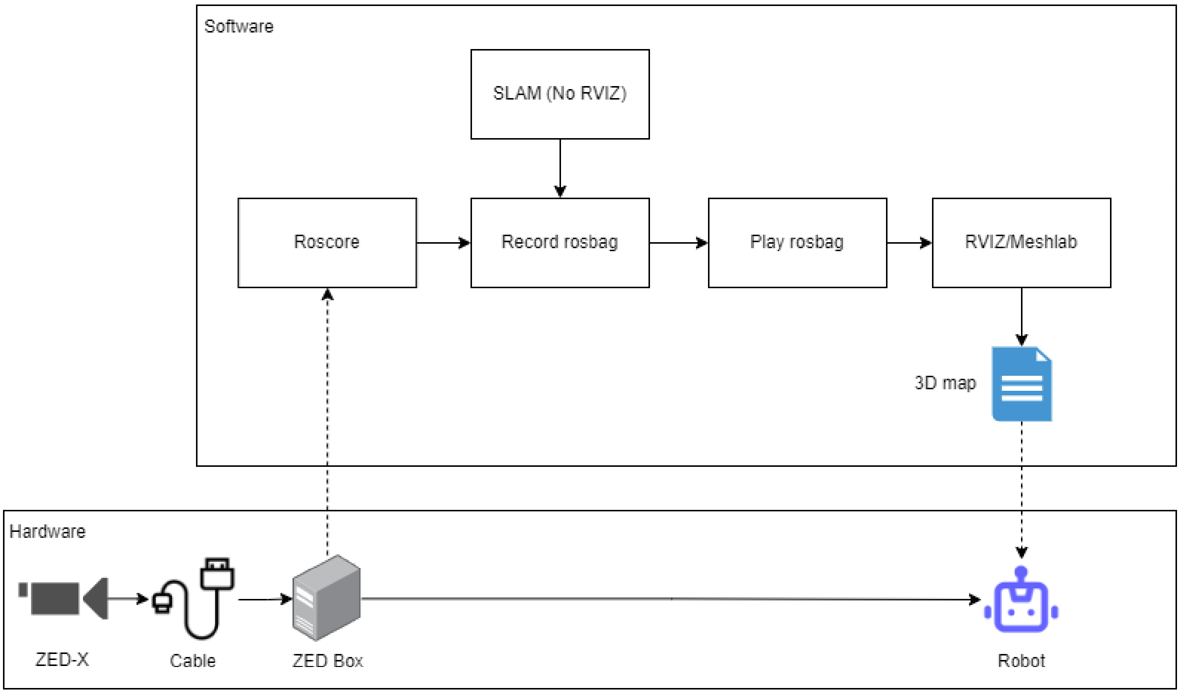

Figure 7. Data flowchart of ZED-X system

3.2.1 Hardware interface of ZED-X system

The following section will introduce the hardware used in the ZED-X system, to get a better understanding of the working and difference between the systems. It also gives a clear view of the computing system and the camera equipment. The technical specifications are organized in Table 2.

Table 2. Hardware specifications of ZED-X experiment

Parameters	**Configuration**
Computing system	
Computer model	ZED Box Orin™ NX
Processor	ARM 8-core Cortex-A78AE
GPU	NVIDIA Ampere 32 Tensor Cores @ 918 Mhz
RAM	16 GB
Camera equipment	
Camera	Stereolabs ZED-X
Connector cable	GMSL2 Fakra 1-to-4 M-F Cable

The GMSL-2 cable connection is a more robust technology for data transfer between devices than a normal USB port. GMSL2 Fakra is designed for high performance in video transmission and can be securely attached to the ZED-X and Jetson. It is a coaxial cable with a forward rate of 3 to 6 Gbps and a reverse data transfer rate of 187 Mbps. The robust design of the cable makes it so the female port can be rotated making it less prone to malfunctions between the devices. The base of the ZED Box is the Jetson Orin NX module with 16 GB of RAM. Jetson is a product line from NVIDIA of embedded boards with Tegra module processors. The Jetson is a low-power consumption system that in the operating system can be set into different power modes. These different power modes allow the board to faster run complex algorithms like Machine Learning or AI applications.

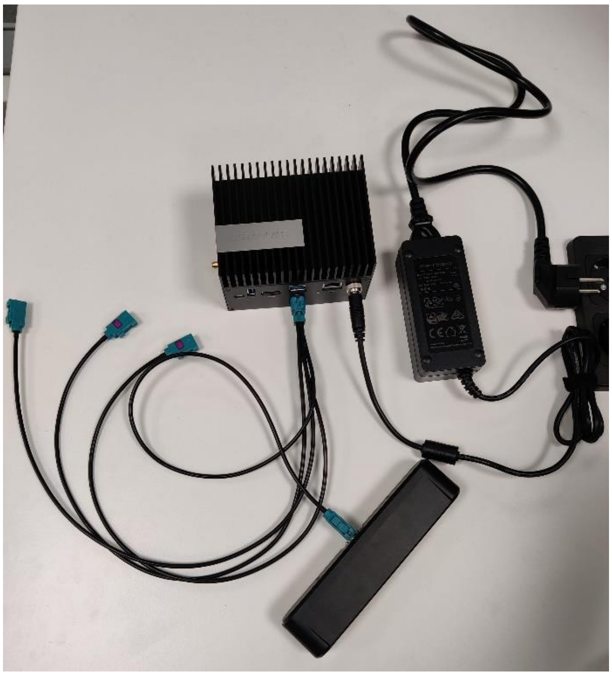

Figure 8. Reconstruction of ZED-X experiment setup

Figure 8 shows the hardware connection as described. The previously mentioned low power consumption mode can be selected in the right top corner of the operating system taskbar. There are options available to run on 5W, 15W and Max (30W) to give power for full computing power available. If the

Jetson is not strong enough on max power the clock speed of the Orin can also be changed to increase performance. In the terminal, the command “sudo jetson_clocks” will change the frequency of the CPU, GPU and EMC clocks.

3.2.2 Software-related matters

The ZED Box runs a modified operating system that is made by NVIDIA for embedded systems. Instead of having the stock version of Ubuntu Linux, it uses the open-source software Linux for Tegra (L4T). L4T is optimized for the Tegra processor chips series providing a smoother experience by using for example the Linux Kernel and file system. Tegra is often used in combination with the NVIDIA JetPack, which is the SDK package provided. This package is designed for the development of AI applications on embedded systems by NVIDIA. JetPack comes with multiple features that accelerate deep learning and neural network processing on Jetson boards. CUDA and NVIDIA drivers as specified in the ZED-2i system are not needed in the ZED-X system. The reason is that JetPack already has preinstalled drivers for the Jetson that enable GPU communication.

3.3 Common software required

Both systems use common software and technologies that are available and compatible with the hardware and operating system of both systems.

3.3.1 ROS Packages and Nodes

As mentioned in the ROS installation section, depending on the implementation, the system needs external packages, also known as system dependencies. In the case of the ZED camera, a specific package is available that includes the nodes that enable the functionalities of the ZED SDK. This package is referred to as "ZED-wrapper," and it allows the camera to provide the following information output through the ZED node: camera left and right imaging, depth map, point clouds, and pose information. These output topics are required as inputs for 3D mapping and localization processing. The SLAM approach utilized the RTAB-Map, a Real-Time Appearance-Based Mapping package. RTAB-Map uses an enhanced loop closure detection method and algorithms to determine new frame images to the corresponding location.

3.3.2 ZED SDK

Stereolabs, the manufacturer of the stereo camera, provides a comprehensive software package designed to enable the camera's complete potential. This service development kit has integrated features and algorithms, which are ready to run and include preset configuration files.

This research focuses on utilizing specific features and functionalities to determine their potential in future applications. These two main elements are positional tracking and spatial mapping, also called SLAM. SLAM is a renowned way for robots to navigate and understand their surroundings.

3.3.3 Other extensions

In configurations regarding software-related matters, other things still need to be considered. One of these is to manage and execute the 3D map data to be retrieved and analyzed after the experiment. There are several ways of doing this within ROS, such as using the RTAB-Map data recorder.

An effective approach involves recording map data during experimental sessions with different parameter files, giving insight into the camera's attributes and limits. The recorded database contains crucial information and can be examined through applications like Meshlab or the RTAB-Map standalone database viewer. Setting up nodes and managing the information and topics being subscribed and published can sometimes be challenging. In response, ROS provides an RQT graph tool that visualizes nodes and topics within the application and how these interact with each other. This tool significantly enhances the understanding of the system and solves communication issues during the tweaking and testing phase.

4 EXPERIMENT SETUP

In this chapter approach of the systems experiments are defined to maximize successful experiment results. The path is made through defining a hypothesis, a goal to reach and the methodology to reach this goal.

4.1 Proposed experiment results

The main reason for starting this research is to gain more information on stereo-based cameras that are enhanced with AI technologies. If the experiment shows that the equipment is suitable for SLAM implementation then it can be installed and used in real-world implementations. The path to this goal can be achieved through actively monitoring the changes in parameters and verifying the impact on the results. The approach of Trial and Error is used to recognize the deviation of the desired outcome. Finding the balance between the highest quality and system limitations requires this experimental technique to be used. Positioning of the equipment is quite tactical as the stereo camera uses depth for sensing distance and localization.

4.2 Focus areas for implementation

The focus of the experiments goes to the integration of the system on a robot. The experiments give insight into the integration and implementation in the proposed environments. In the proposed environments there are a lot of details and objects to detect by the depth sensor. The depth sensor and point cloud registration are used to show higher quality feedback in environments with a high quantity of small objects and structures and low quality with smooth long objects. Assuming that both systems perform this way, the experiments are focused on the indoor environment with both smooth and small objects. The reason for not doing an outdoor comparison is also related to the weather conditions at the time of doing the experiments. The safety of the hardware cannot be guaranteed, which leaves indoors as an option. Comparison of the results after doing the experiments the following as a factor: Accuracy, depth sensing and object detection.

5 RESULTS OF SYSTEM EXPERIMENTS

The experiment setup with the outlined configuration in Chapter 3 has been deployed for several tests in different environments. The configuration files have been adjusted to optimize performance according to the surface structure of the area in this specific environment. Fine-tuning of the files has led to notable improvements in outcomes.

5.1 Experiments of ZED-2i system

The ZED-2i system has been executed two times in different environments to gather enough data for an initial benchmarking. The experiments are divided into two sub-sections labeled as indoor and outdoor.

5.1.1 Indoor experiment of ZED-2i system

Execution of the indoor experiments involves office spaces, corridors, and an electric laboratory, all by walking through the area or stationery 360-degree camera rotation. Note that the initial round of indoor tests had a less advanced node, package, and parameter setup script.

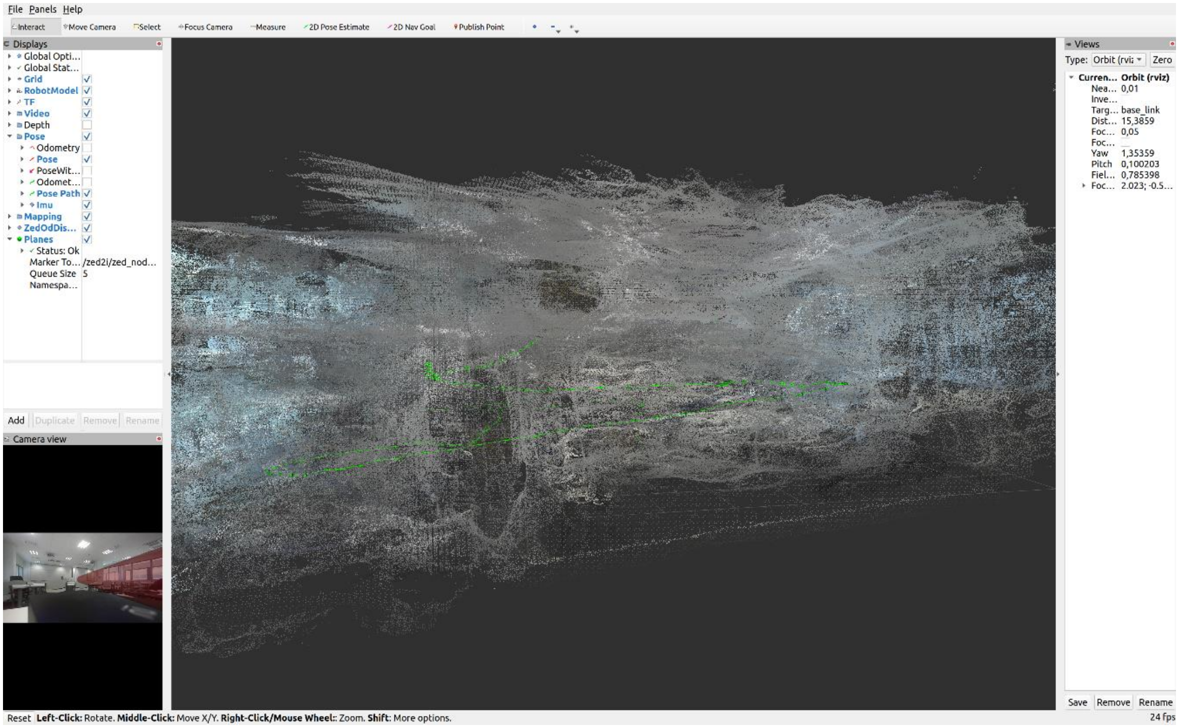

Figure 9. First indoor experiment ZED-2i

The first experiment, as captured in Figure 9 shows the initiated primary results with many challenges concerning cloud density and surface detection. However, larger structures like pillars, doors, and windows remained easier to recognize. Individual point clouds did not display the correct color coding and showed little or no clustering. After the first experimental experience, a modified version of nodes and parameters is prepared to be deployed. The modifications to the configuration are aimed at eliminating the limitations of the first experiment.

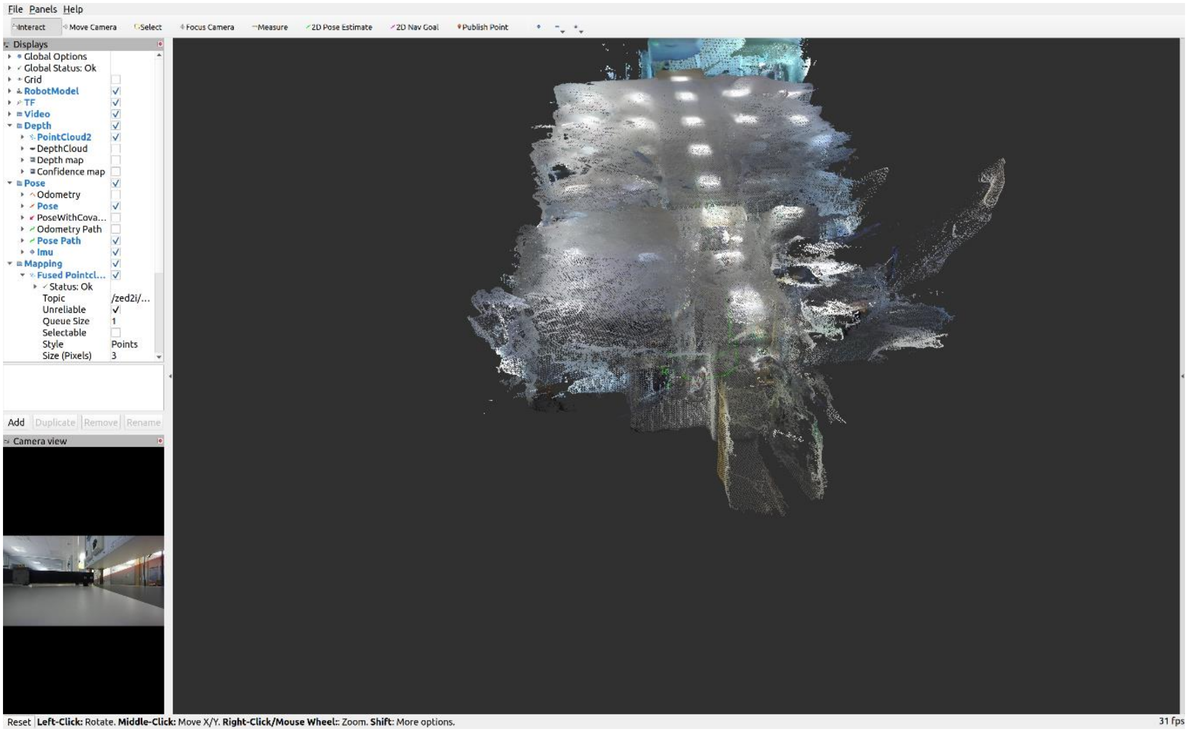

Figure 10. Second indoor Experiment ZED-2i system

Figure 10 presents a more advanced cloud clustering and color grading level. The camera captures objects with more detail, such as square light pods in the ceiling, wooden doors, and walls. These objects have a higher cloud cluster density than the earlier example. The outcomes are outstanding around brighter areas, for instance, emergency exit signs and within the back room, where sunlight comes through the windows. Besides that, it faces challenges detecting several larger objects like workbenches.

Continuing the indoor test run, the camera is rotated and moved minimally along the x and y-axis, making it almost stationary. This approach accentuates the difference and ability of the camera to effectively identify smaller objects and differentiate colors in the environment, as illustrated in Figure 11.

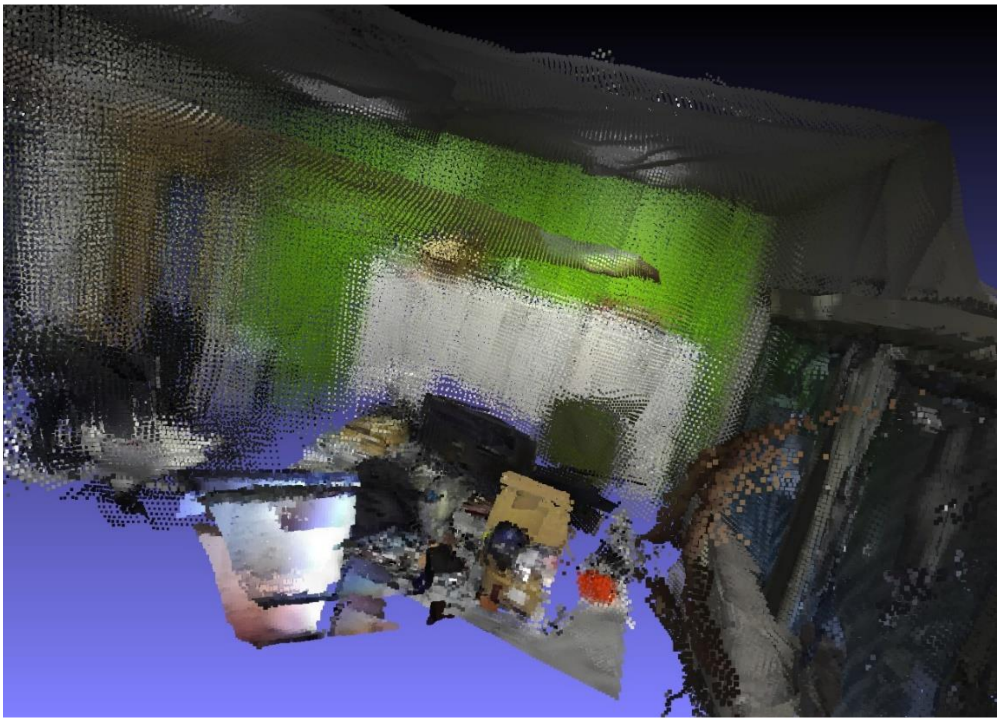

Figure 11. Indoor point cloud ZED-2i system

Even surfaces like cabinets and a flat green wall, computer monitors, boxes, desks, and chairs can be identified. Certain clouds are incorrectly picked up or duplicated in positions like the monitor and a brown bar extending from the door. The RGB picture of the environment shown in Figure 12 visualizes and emphasizes the quality of the results.

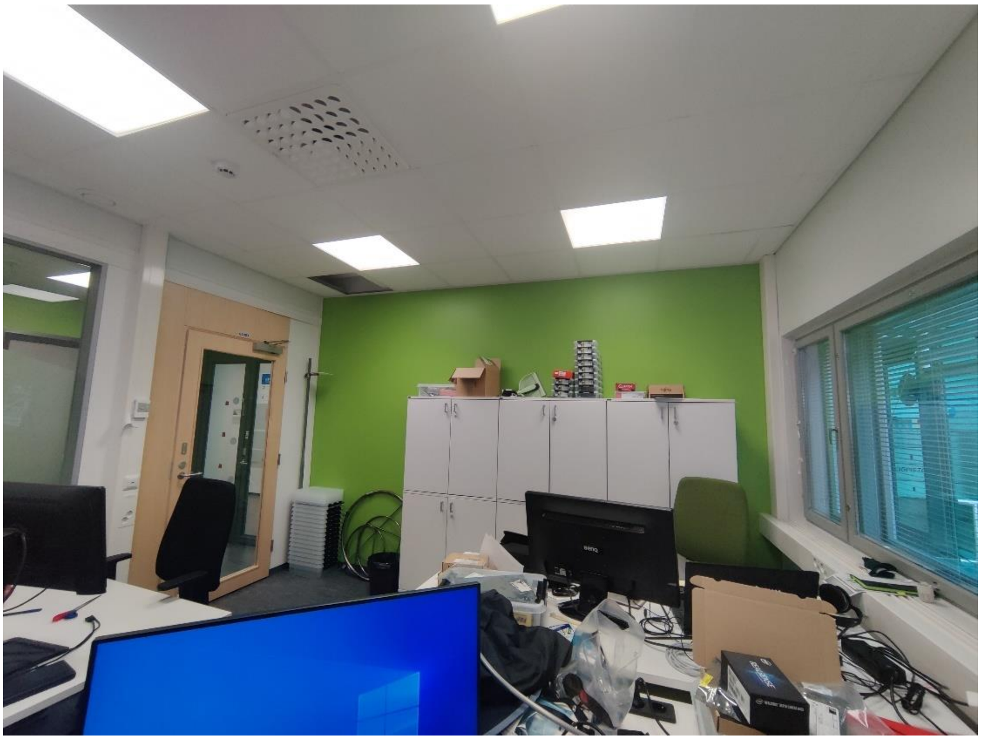

Figure 12. RGB picture of experiment environment of ZED-2i system

5.1.2 Outdoor experiment of ZED-2i system

Outdoor experiments are conducted by walking and traversing a dirt road with plenty of bushes and trees on either side. The capturing process is initiated by the launch file at the start of the road until a particular turning point before returning to the initial position. The camera is manually carried and positioned approximately 1.5 meters above the ground. Figure 13 illustrates the upper perspective of the resulting outcome.

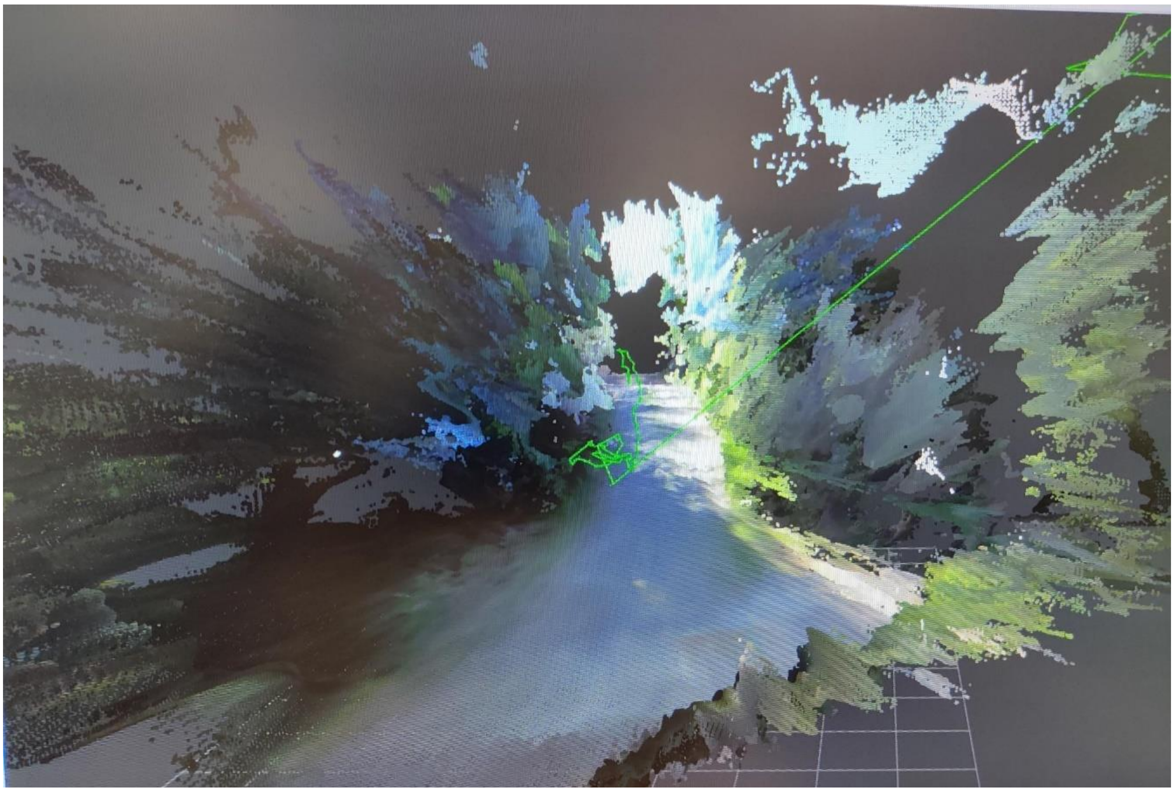

Figure 13. Outdoor point cloud ZED-2i system

From an elevated viewpoint, the shade pattern from trees becomes distinctly noticeable on the road. The shade is of precise shape that matches the branch that obstructed sunlight on the ground. It is worth mentioning that there is noise in detecting the sky because of the bright sunlight reflection in the camera. This occurrence is observed mainly in the upper middle and upper right sections. The issue could potentially be linked to either camera polarization or the angle of sunlight making it too bright to visualize the correct colors. Throughout the recording process, the shaping of objects via point cloud mirrors real-world observations surprisingly well.

Upon reviewing the results, an issue becomes visible regarding the layering of the environment based on the camera's visual odometry-based localization method. The background was captured as intended up to the turning point, as visualized in the lower section of Figure 14.

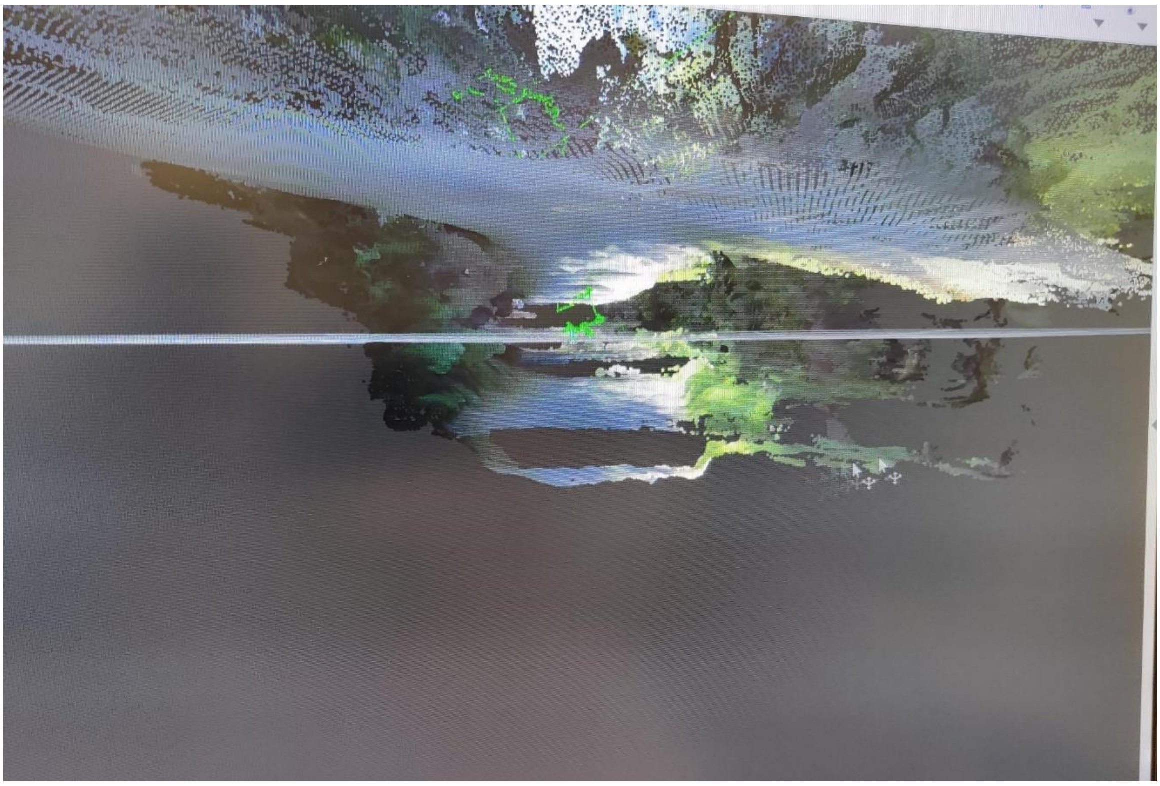

Figure 14. X-axis view of outdoor experiment with ZED-2i system

At the point of turning, a shift on the y-axis was recorded, leading to an additional layer being built on top of the original. The quality of the first half of the experiment, the lower section, contains a higher clarity and increased cloud density. In conflict, the upper section has a lower density, and clouds are scattered around and do not form recognizable objects or structures.

5.2 Experiment of ZED-X system

The ZED-X system experiment has been executed a few times in an indoor environment to collect the optimal parameters for the selected environment. In the ZED-X system, some challenges occurred, which will be discussed in the next sub-section.

5.2.1 Hardware issue ZED Box

During the extensive testing of the ZED-X system the ZED Box started operating differently than it supposed to do. The ZED Box was restarting the operating system in the middle of an experiment making it unable to retrieve the data. The issue was analyzed but a solution was not found till it started boot looping. The supplier was contacted regarding the issue and based on the description a replacement for the ZED Box was issued. Feedback on the malfunctioning ZED Box from Stereolabs was that there was a hardware issue. The process of sending and receiving the replacement took a month delaying the development and testing of the ZED-X system. In this process, the data on the malfunctioning ZED Box couldn't be retrieved and a new installation and workspace had to be created.

5.2.2 Other issues encountered in the preparation phase

The first experience of running SLAM on the ZED Box was not successful. There were multiple errors related to the connectivity of the ZED-X camera and to the processing speed of the ZED Box. Whenever the script has established and opened the connection with the ZED-X before, the camera will not be detected by the operating system. Detection of the camera can be verified through one of Stereolabs SDK tools called ZED Diagnostic. If this tool shows that the connection is not properly made the diagnostic report has to be analyzed by putting the following into the terminal: "sudo ZED_Diagnostic --dmesg". In the report, the camera is detected but seems not to respond because it is not closed correctly. Closing the connection is done on termination of the experiment, but not registered by the operating system itself. The solution for this bug is to restart the daemon with the following command: "sudo systemctl restart zed_x_daemon".

5.2.3 ZED Box export file conversion

After receiving data and establishing the connection with the ZED-X camera, the terminal showed multiple warning messages regarding low frame rates. The low frame rate can be caused by limiting computational power supplied by the GPU, CPU or RAM. In the case of ZED Box the GPU has more suitable power to do the calculations than the ZED-2i system. The bottleneck in the ZED-X system is most likely related to the CPU or RAM available. Result of the bottleneck in the system the experimental data had to be collected and visualized later. There are several ways of doing this for example using the rosbag function or collecting the values in a PCD file. The rosbag records the topics selected and can be played and visualized separately. If the PCD file is extracted it has to be converted to a PLY file type to be used as an input for Meshlab. In this experiment, the rosbag has been used because this was a suitable and fast option to generate results.

5.2.4 Indoor experiment of ZED-X system

Execution of the ZED-X experiment was done in the same office space as the ZED-2i system to create an understanding of the settings and parameters for this system. The experiment trial starts collecting data while it is being rotated from left to right.

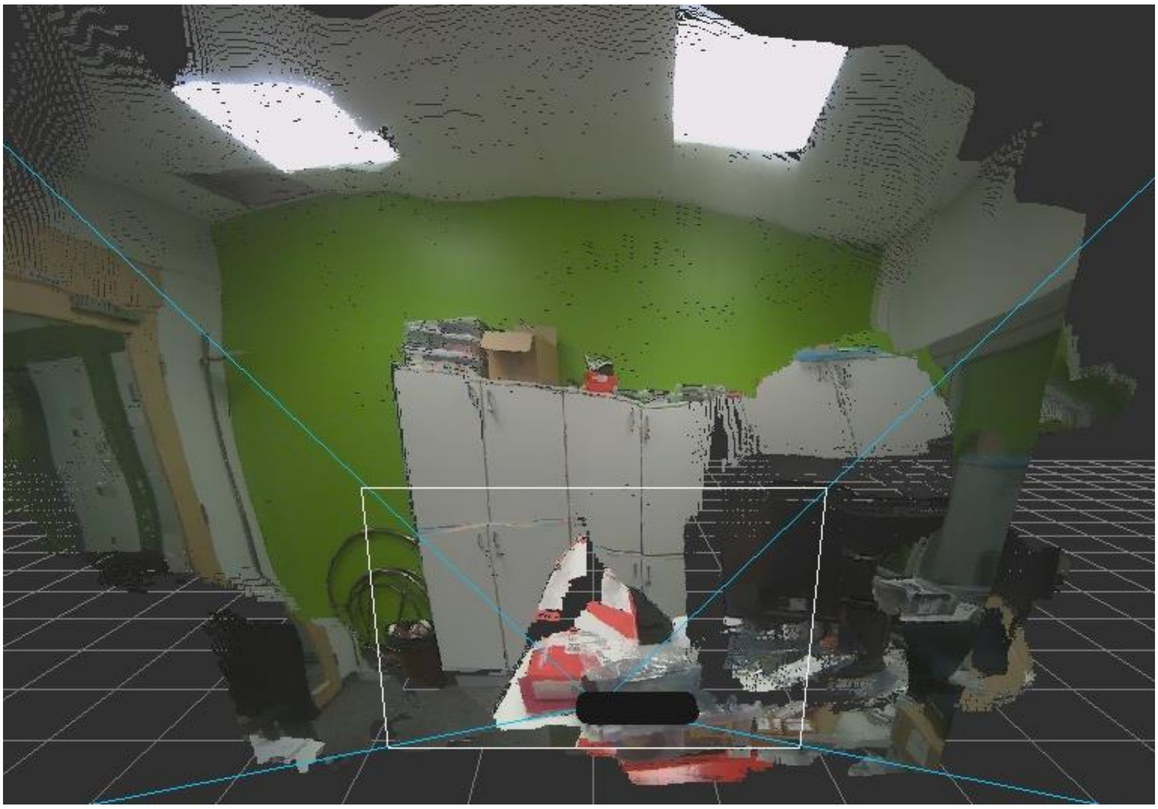

Figure 15. First ZED-X experiment

The first recovered experiment results suitable to be analyzed are captured in a snapshot in Figure 15. The snapshot of the map created with the ZED-X data is of super high quality making it similar to a RGB picture from a distance. The point cloud density and depth sensing accuracy make objects almost as smooth as a mesh. If zoomed in closer to the objects the individual point clouds are clearly visible. Positives are noticed on the green wall and around the white cabinet detecting the depth of the edges very well. On the right side of the picture, the monitor is not detected in the map, because the camera was not in the position to get this area properly during the experiment.

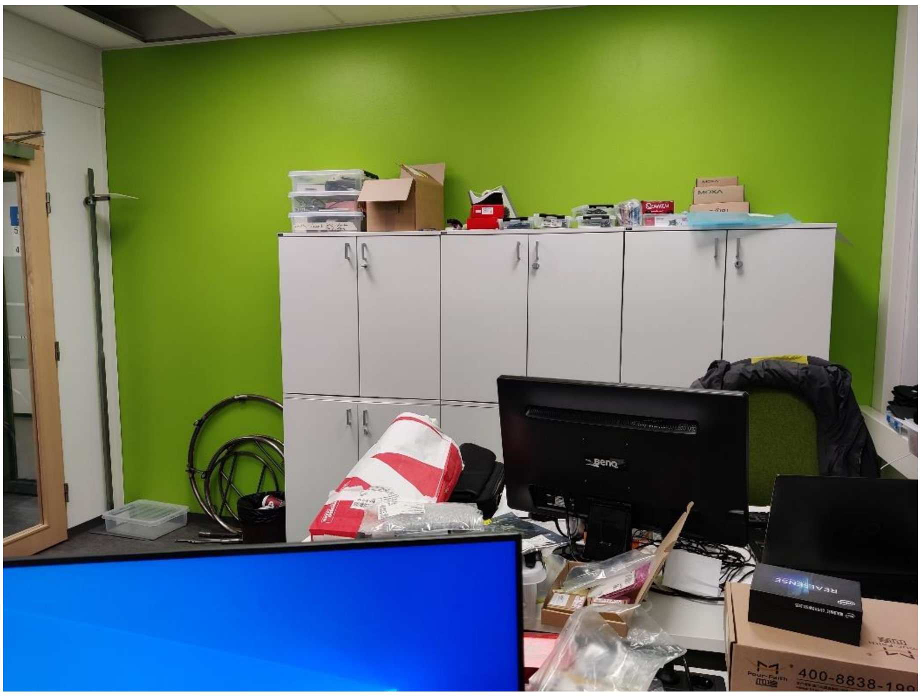

Figure 16. RGB picture of experiment environment of ZED-X system

In Figure 16 an updated representation of the office environment can be seen. Comparing the RGB pictures from both systems, the main difference is that boxes were moved and there are some brighter colors to be seen.

On the right side of Figure 15, the map looks distorted for some reason. Analyzing the depth map gave more insight into the possible problem with the camera. Figure 17 shows the depth map of the camera while recording the experiment. Colors in the depth map of the ZED-X represent the distance from the camera also known as depth. Red is the closest to the camera, followed by green and blue. Black areas are not properly detected or too far meaning the confidence level is below the threshold of 95.

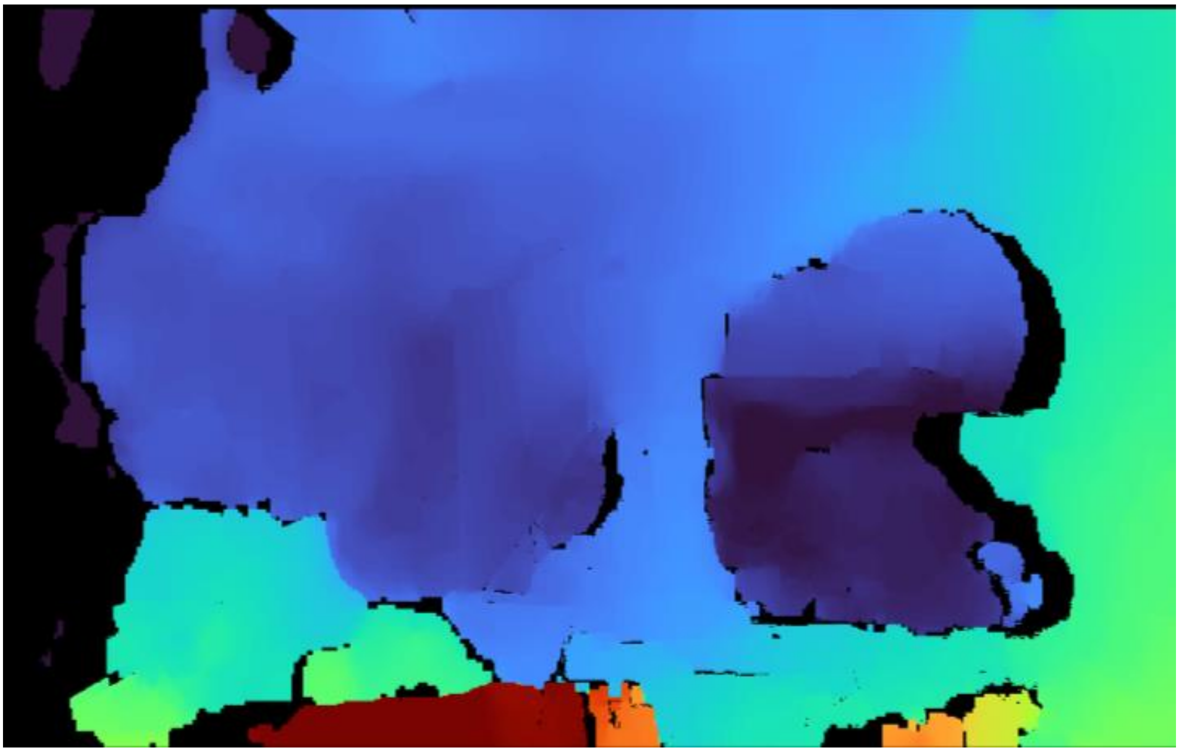

Figure 17. ZED-X depth map

In the snapshot it looks like a dirty spot on the lens of the ZED-X changing the point clouds received. The lens has been cleaned and clear of any possible obstructions, but the depth map stays the same. In the next experiment with this particular device, the depth detection has to be analyzed to get more accurate results. In the interpretation of the received results in this experiment, the obstruction on the right side of Figure 17 is not the main focus. Whenever the map is zoomed in on the left side the point clouds are clearly visible, as expected in Figure 18.

Figure 18. ZED-X zoomed view

6 ANALYSIS OF THE RESULTS

Chapter 6 will be used to analyze the results in sub-sections to clarify the strengths and challenges of each system. In this section also the comparison is made by looking at the quality perceived.

6.1 Hardware performance of the systems

Both systems performed well under the high and demanding algorithms and settings used in the experiments. While using the maximum parameters the systems did not fail nor crash during the recording and optimizing phase. The ZED-2i system was easy to use and the camera’s maximum resolution could be selected at a high enough frame rate to visualize it in real time on the same device. The ZED-X system was struggling with the processing of the data while visualizing it at the same time. Warning messages were prompted in the terminal continuously saying the frame rate couldn’t be handled in time. The frequency and queue size were changed multiple times but this didn’t solve the issue.

6.2 Perceived quality of the systems

Looking at the overall quality and condition of the analyzed results the ZED-X seemed to have out-performed the ZED-2i. This result was expected by considering only at the technical specifications of the systems. The ZED-X system has a more powerful GPU available in the ZED Box and the ZED-X stereo camera has a higher resolution and more accurate depth sensing. It is worth mentioning that the libraries in ROS haven’t been optimized for the newest JetPack version running on the Jetson Orin NX. There are still improvements to be made in this area to increase the gap between the two devices.

7 DISCUSSION AND CONCLUSION

In this final chapter the conclusions of the research, discussion and possible future research are explained.

7.1 Conclusion

In conclusion, the ZED-X in combination with the ZED Box has shown more enhanced results by using point cloud and SLAM. The ZED-X system showed no false detections and has a high resolution making it an effective system for implementation on a robot. Even while the ZED Box was limited in available computational resources it outperformed the other system. The ZED-2i system has also proven to be a solid solution for basic usage in less critical situations, because of the deviation in the experiment results. In this thesis, a clear overview of the differences in working of the systems and possible implementation fields are given. The AI-enhanced stereo camera proves to fulfill a critical role for navigating and positioning a robot in an unknown environment. It displays to be an effective solution under different conditions with the use of SLAM.

7.2 Discussion

Technologies such as SLAM are important for AI-enhanced cameras to be used in autonomous robots. However, in the case of this thesis, the libraries were not optimized for the use of the newest hardware and operating systems installed and available for testing. The emphasis lies on the potential of collecting better results following optimal tweaking of variables giving a more grounded comparison. Through extensive refining of the models, RVIZ could be used in visualizing and observing the collection and forming of the 3D map in real-time. This can give a better indication of what is being registered correctly or incorrectly and what is being skipped.

Furthermore, the limited time for testing the ZED-X camera did make it difficult to rule any of the systems out of being implement-worthy. Understanding of the performance and value both systems bring depending on the use case is the core of this thesis.

REFERENCES

Altuntaş, N., Uslu, E., Çakmak, F., Amasyalı, M., & Yavuz, S. (2017). Comparison of 3-dimensional SLAM systems: RTAB-Map vs. Kintinuous. *2017 International Conference on Computer Science and Engineering (UBMK)* (pp. 99-103). Antalya: IEEE.

Chiodini, S., Giubilato, R., Pertile, M., Tedesco, A., Accardo, D., & Debei, S. (2021). Design of a user-friendly control system for planetary rovers with CPS feature. *8th International Workshop on Metrology for AeroSpace (MetroAeroSpace)* (pp. 317-321). Naples: IEEE.

Dave Hershberger, D. G. (2023, November 16). *rviz Documentation*. Retrieved from rviz: Main Page: https://docs.ros.org/en/noetic/api/rviz/html/c++/index.html

de Jesus, K. J., Kobs, H. J., Cukla, A. R., de Souza Leite Cuadros, M. A., & Gamarra, D. F. (2021). Comparison of Visual SLAM Algorithms ORB-SLAM2, RTAB-Map and SPTAM in Internal and External Environments with ROS. *2021 Latin American Robotics Symposium (LARS)* (pp. 216-221). Natal: IEEE.

Fernandez, L., Avila, V., & Goncalves, L. (2017, May 23). A Generic Approach for Error Estimation of Depth Data from (Stereo and RGB-D) 3D Sensors. *Preprints*, p. 12.

Firdaus, A. R., Hutagalung, A., Syahputra, A., & Analia, R. (2023, January 13). Indoor Localization Using Positional Tracking Feature of Stereo Camera on Quadcopter. *Electronics, Unmanned Aircraft Systems with Autonomous Navigation*, p. 12.

Flyability. (n.d.). *WHAT IS SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)?* Retrieved December 12, 2023, from Flyability's Confined Space Drone Makes Inspections Safer & Cheaper: https://www.flyability.com/simultaneous-localization-and-mapping

J. Pena Queralta, F. Y. (2019). FPGA-based Architecture for a Low-Cost 3D Lidar Design and Implementation from Multiple Rotating 2D Lidars with ROS. *IEEE Sensors* (p. 4). Montreal: IEEE.

LucasWalter. (2019, September 5). *roscore - ROS Wiki*. Retrieved from Documentation - ROS Wiki: https://wiki.ros.org/roscore

MathWorks. (2023, November 13). *Perform SLAM Using 3-D Lidar Point Clouds*. Retrieved from MathWorks - Makers of Matlab and Simulink: https://www.mathworks.com/help/nav/ug/perform-lidar-slam-using-3d-lidar-point-clouds.html

Ramachandran, S., & Sahin, F. (2019). Smart Walker V: Implementation of RTAB-Map Algorithm. *14th Annual Conference System of Systems Engineering (SoSE)* (pp. 340-345). Anchorage: IEEE.

Rohit Yadav, R. K. (2022). Fusion of visual odometry and place recognition for SLAM in extreme conditions. *Springer Nature*, 1-20.

ROS Topic command line tools – practical example (rostopic and rosmsg). (2023, December 15). Retrieved from Learn Robot Programming - The Robotics Back-End: https://roboticsbackend.com/ros-topic-command-line-tools-practical-example-rostopic-and-rosmsg/

Tadic, V. (2023, February 10). Study on Automatic Electric Vehicle Charging Socket Detection Using ZED 2i Depth Sensor. *Electronics, Advances in Image Processing, Artificial Intelligence and Intelligent Robotics*, p. 24.

Wolf, Á., Troll, P., Romeder-Finger, S., Archenti, A., Széll, K., & Galambos, P. (2020, October 8). A Benchmark of Popular Indoor 3D Reconstruction Technologies: Comparison of ARCore and RTAB-Map. *Electronics, Computational Cybernetics*, p. 9.

Zhang, G., Zhisheng, Z., Zhijie, X., Min, D., Meng, P., & Cen, J. (2022). Implementation and research on indoor mobile robot mapping and navigation based on RTAB-Map. *28th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)* (pp. 1-6). Nanjing: IEEE.