Development of a Mobile LiDAR for Surveying Applications with Assessment of its Accuracy

by

Abdallah Ahmad Adnan Taha

A dissertation presented in partial fulfillment of the requirements for the degree of Master of Science in the Department of Geomatics, Computer Science and Mathematics, Stuttgart University of Applied Sciences

Declaration

The following Master thesis was prepared in my own words without any additional help. All used sources of literature are listed at the end of the thesis.

I hereby grant to Stuttgart University of Applied Sciences permission to reproduce and to distribute publicly paper and electronic copies of this document in whole and in part.

Stuttgart, 28.02.2023

Abdallah Ahmad Adnan Taha

Approved by:

Prof. Dr. Paul Rawiel

ACKNOWLEDGEMENT

"In the name of Allah, Most Griseous, Most Merciful"

The work in this thesis could not have been accomplished without the cooperation of other individuals. Professor Paul Rawiel, my supervisor from HFT Stuttgart University, and Professor Amjad Hawash, my co-supervisor from An-Najah National University, are positioned on top of them. Their persistent support and encouragement, along with their wise counsel and suggestions, have always helped to create a favourable research atmosphere that has led me in the right direction. For this, I am sincerely grateful and appreciative.

My sincere respect and appreciation are also conveyed to my family, starting with my father, Dr Ahmad Taha, who has been by my side in every stage of my life, beginning with financial support and ending with concern for my future. In addition, I would like to thank my mother Rawda, my sisters Hanin and Razan, and my brothers Yousof, Hamza, Rayan, and Rida for their support during this difficult time. Your encouragement, confidence in me, and moral support have always inspired me to achieve my academic ambitions despite our physical separation.

I would like to express my gratitude to The German Academic Exchange Service (DAAD) for providing me with a scholarship to pursue a Master's degree in geomatics in Germany. I would also want to thank AN-Najah National University for nominating me for this scholarship.

Special thanks go to every member of Agel family (my second family in Germany): Amal, Fayeq, Junes, and Thaer. Your unlimited support to me, never forgettable. Also, many thanks go to Al-Beik family, who were supporting me during my study in Germany.

I am appreciative of the unwavering support and desire of all the Master Photogrammetry and Geoinformatics lecturers to make sure that our master's studies are great. Also, I would want to express my gratitude to all my fellow students in the master's program for their great company and friendship, especially Mohammad Al-Fakhori, Archana KC, Mohammad Al-Khatib, Yaha Saydawi, and Fahed Al-Saleh.

I've been blessed to have a lovely supporting family throughout my entire life, and I'd like to take this moment to thank my mother, father, brothers, and sisters for their unflinching support of my education and research.

I dedicate this thesis to my amazing family.

Assessment of Accuracy of a Mobile LiDAR Developed for Surveying Applications

ABSTRACT

Surveyors used to collect precise data utilizing reliable tools including static laser scanners, Total-Station, and Global Navigation Satellite System (GNSS) receivers. Each of these technologies has limitations, though. While some of them, like static laser scanners, are quite expensive, others rely on single-point data collection (i.e., GNSS instruments and Total-Stations). Recently, several companies have created portable handheld Light Detection and Ranging (LiDAR) systems that can be used for mapping and surveying tasks based on Velodyne's Puck VLP-16 LiDAR sensor. In addition to being lightweight mobile equipment (portable devices) that are convenient to use in the field, LiDAR can quickly acquire millions of 3D point clouds. They are portable, lightweight, and simple to use in the field. However, the overall cost of such device is around €30,000 or more.

There are two aims of this research: first one is to develop a low-cost Mobile LiDAR RTK (MLRTK) system (of about €5,000) integrated with a Real Time Kinematic (RTK) GNSS receiver to be used for data collection of different surveying applications. Second aim is to develop a software that is capable of integrate RTK trajectory with Simultaneous Localization And Mapping (SLAM) trajectory to produce a corrected trajectory which can be then used to build a precise 3d map. In addition, the accuracy of the proposed systems will be evaluated.

On one hand and after extensive research, it was possible to combine and to configure different sensors to create the MLRTK device with a cost of about €3,500. The created gadget was successfully used in the field to collect both LiDAR and GNSS RTK data. On the other hand, this research also led to the development of a LiDAR RTK program. Via a series of interpolation, smoothing, and transformation stages, this software successfully combines RTK and SLAM trajectory data to provide a corrected trajectory that can then be used to produce three-dimensional maps. In addition, the evaluation of the LiDAR data shown centimetre level of accuracy obtained in both relative and absolute accuracy assessment tests. It is important to note that one of the research's most important findings is the ability to use the MLRTK device and software created within this research to bridge the GNSS outage gap.

Keywords: Surveying, LiDAR, GNSS, Mobile System, Hardware Integration, SLAM, LiDAR Software.

TABLE OF CONTENT

TABLE OF FIGURES

TABLE OF TABLES

ABBREVIATIONS

2D	Two-Dimensional
3D	Three-Dimensional
AOI	Area Of Interest
GPS	Global Positioning System
GNSS	Global Navigation Satellite System
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
LiDAR	Light Detection And Ranging
MLRTK	Mobile LiDAR RTK device
NMEA	National Marine Electronics Association
NTRIP	Networked Transport of RTCM via Internet Protocol
PCAP	Packet Capture
PCD	Point Cloud Data
RTK	Real-Time Kinematic
SLAM	Simultaneous Localization and Mapping
ToF	Time of Flight
TS	Total Station

1 INTRODUCTION

1.1 Contextual Background

Since LiDAR can precisely acquire large number of measurements quickly, it makes it one of the promising technologies for future surveying applications. Traditional surveying methods are not only time-consuming but are also limited in their ability to capture rugged terrain or smaller spaces due to single point-based data collection. In addition, static laser scanners can measure millions of points in a short time, but these instruments have a stationary structure and are very expensive devices. Recently, hand-held three-dimensional (3D) LiDAR systems have significantly reduced survey time when compared to static laser scanning and fieldwork methods. In addition, these systems offer a potential solution to the problem of creating ground-based point clouds with the necessary geospatial extent whilst maintaining accuracy [1]. Moreover, these devices become available at affordable prices, such as Velodyne's Puck VLP-16 LiDAR sensor. This LiDAR has 16 channels and can generate 300,000 points per second from a 360° horizontal field of view and a 30° vertical field of view with a 15° elevation from the horizon[1]. The data from LiDAR is measured as frames and can be combined in a process called registration to produce a map using either the SLAM technique or using GNSS in combination with the Inertial Navigation System (INS). Using a SLAM algorithm, LiDAR can produce a precise map of the surrounding environment [2]. Additionally, SLAM mainly obtains the pose by matching the observed environmental features with the feature map while moving and simultaneously updating the feature map to achieve autonomous positioning [3]. It can determine the user's location as a 3D position or as a 3D trajectory path in the local coordinate system. However, the GNSS / Inertial Navigation System (INS) system uses a tac-tical Inertial measurement unit (IMU) that costs several thousand euros and can be used with LiDAR to produce a georeferenced map. Several businesses have recently created a portable mobile LiDAR system for use in surveying projects and on building sites to generate 3D maps, topographic maps, and other maps. These instruments have proved to be powerful instruments for rapid data collection for different surveying projects. This research will examine the ability to develop a mobile LiDAR system integrated with an RTK GNSS receiver, which can be used for data collection for surveying applications. The data from this device will first be post processed with SLAM based software. Then it will be post-processed a second time based on RTK GNSS trajectory data to produce a georeferenced map. The resultant map from this system will be compared with a pre-surveyed map for accuracy assessment.

1.2 Problem Statement and Motivation

Surveyors are looking for a low-cost hand-held sensor that can be used for the determination of their location and for mapping the surrounding area. Such device can cost about €30,000 due the use of LiDAR system integrated with high-performance tactical IMU. However, constructing similar system presents several difficulties, beginning with combining the correct components and ending with configuring them to function as one device. However, the success of developing the proposed MLRTK system will pave the way to producing a commercial low-cost surveying device.

Managing the outcome data from the sensor is additionally difficult due to the lack of available free software that can combine LiDAR data and GNSS RTK data. The available commercial software can process LiDAR and INS together. Therefore, it is necessary to develop a software that can deals with available LiDAR and GNSS RTK data. As a result, surveyors can use the proposed system on construction sites as well as to create a variety of surveying maps, such as 3D maps, topographic maps, and as-built maps.

1.3 Research Aim and Objectives

Mainly, there are two aims of this research: the first one is to create a low-cost mobile LiDAR RTK (MLRTK) system (about €5,000) that is integrated with a RTK GNSS receiver and can be used to collect data for various surveying applications. The second aim is to create a piece of software that can combine RTK trajectory with SLAM trajectory to create a corrected trajectory that can be utilized to create an accurate 3D map. Also, the proposed systems' accuracy will be assessed. To achieve these aims, the following specific objectives have been drawn:

- Review of the scientific principles and historical research behind the technologies.
- Build a mobile LiDAR system through hardware integration.
- Developing a software package to read the LiDAR Packet Capture (PCAP) data, to extract the RTK GNSS observations, and to produce a correct trajectory data based on RTK GNSS data.
- Evaluating the relative accuracy and absolute accuracy of the output data from the device within a pre-surveyed map.

This study seeks to answer the following questions:

- What are the components needed to build a low-cost MLRTK device?

- What is the overall cost of this device?
- Can the instrument be utilized easily to collect outdoors data?
- Could the result be translated into a real-world coordinate system?
- Is the final product suitable for surveying and capable of replacing a traditional GPS?

1.4 Proposed Methodology

The workflow of the proposed methodology is illustrated in Figure 1.

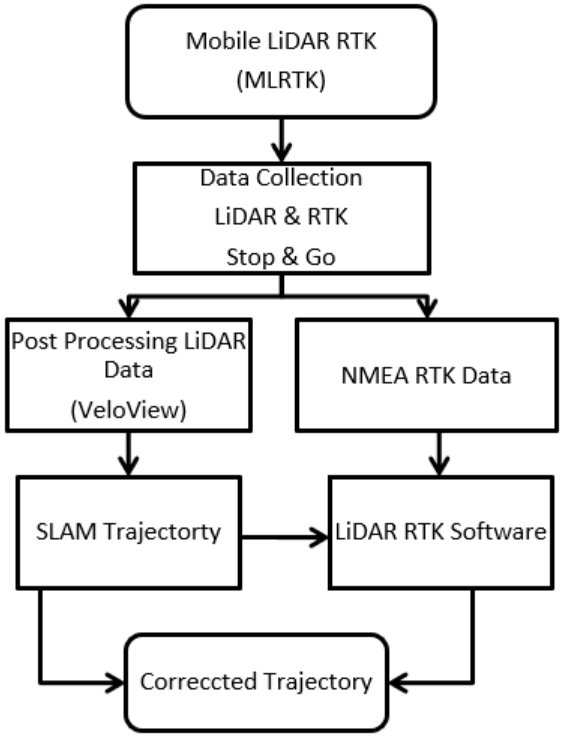


Figure 1: An overview of the workflow of the proposed method.

The proposed methodology of the research is as follows:

- Develop a low-cost MLRTK device through hardware integration.
- Data collection using the developed device to collect LiDAR data and RTK data in stop and go technique. Data collection started from a specific point with static data collection, then moving to another point and collect again static data and so on to close the loop on starting point.
- Post-process LiDAR data using an open-source software (VeloView) to produce a SLAM Trajectory data.

- Developing a software package to combine SLAM Trajectory data with GNSS RTK data as follows:
    - Read the LiDAR PCAP file and extract LiDAR and positional data.
    - Add RTK data from National Marine Electronics Association (NMEA) file and update positional data.
    - Add Slam trajectory data.
    - Combine RTK trajectory data and SLAM trajectory data to produce a corrected trajectory data.

On the one hand, closed loop measurements have proven that the precision of GNSS RTK data can reach centimeter-level of accuracy. On the other hand, SLAM trajectory and a 3D point cloud are produced by post-processing LiDAR data. Unfortunately, there are various errors in the SLAM trajectory, particularly in an outdoor environment. When high accuracy RTK data and SLAM trajectory data are combined, a corrected trajectory and consequently exact 3D maps are created.

1.5 Study Area

There are two study areas that utilized in this research: one in Palestine, and the other in Germany. The Palestine study region was the site of the outdoor test, whereas the German study area was the site of the indoor test. This is due to a drone does not require a particular permit to map the Palestine test region.

The Palestine Area of Interest (AOI) was chosen nearby the author's home in Nablus, West Bank, and the center point's coordinates were as follows: latitude: 32.211188°N, longitude: 35.243845°E. This AOI is approximately 200 meters by 100 meters, covering an area of 22,000 square meters Figure 2. To serve as testing points with known locations, there were 6 points painted as crosses on paved roads of the AOI (Figure 2). While the road between points 4 and 6 is concave, the road between points 1 and 3 is nearly level. Moreover, there was an elevation difference of up to 7 m and 3.6 m between points 3 and 4 and 6 and 1, respectively. The research area consists of houses, roads, street-lamps, electrical pools, scattered trees, and elevated parcels of land.

Fehler! Verwenden Sie die Registerkarte 'Start', um Heading 1 dem Text zuzuweisen, der hier angezeigt werden soll.

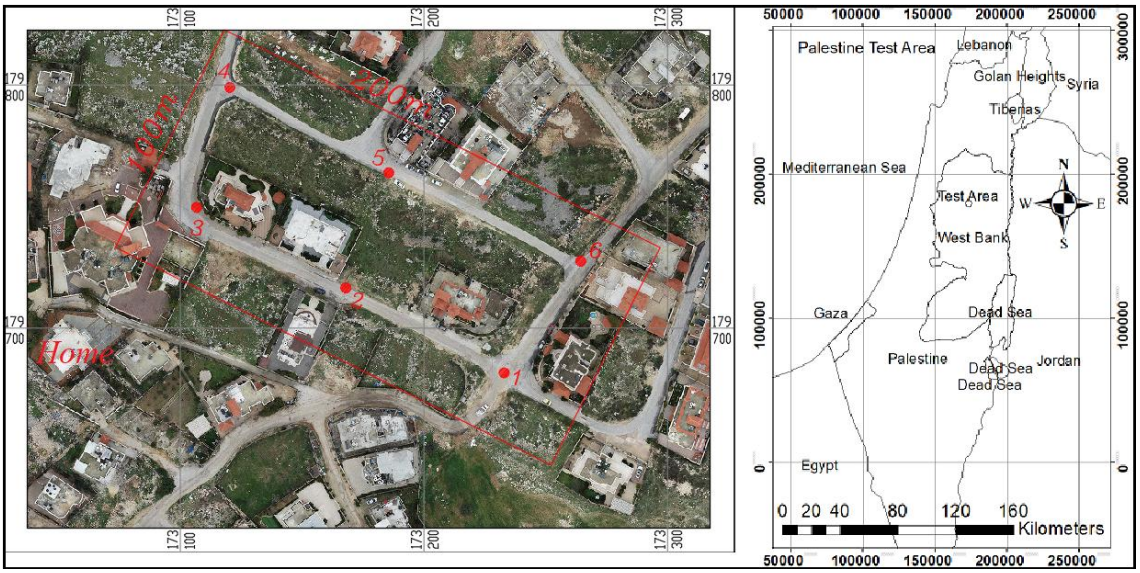


Figure 2: (Left) Palestine Area Of Interest (Right) Map of West-Bank, Palestine.

The German AOI is the ground floor of HFT-Stuttgart's Building 1 in Stuttgart, Germany Figure 3. The approximate coordinates of the AOI with coordinates of latitude: 48.78051°N, longitude: 9.17296°E. This AOI is approximately 24 meters by 19 meters, covering an area of 456 square meters. Doors, walls, windows, and tables constitute the research area.

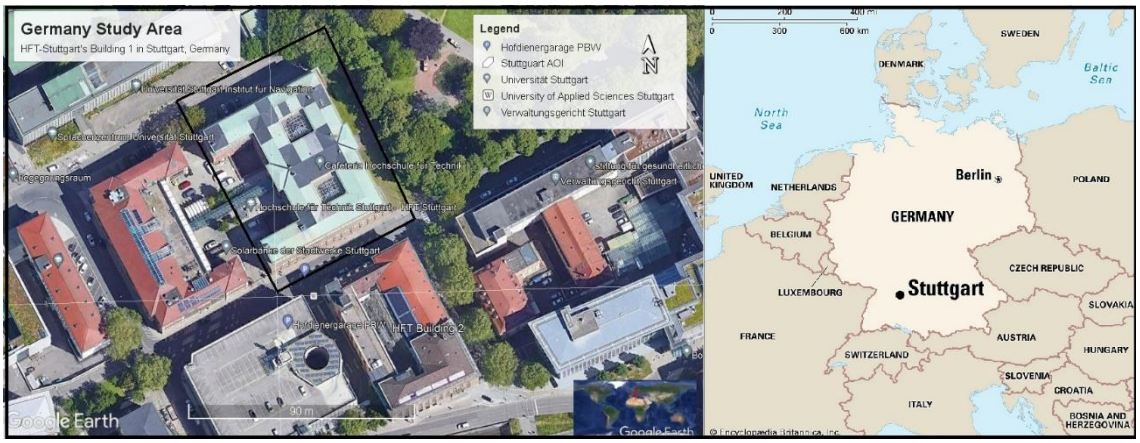


Figure 3: (Left) Germany Area Of Interest (source: Google Earth Pro) (Right) Map of Stuttgart, Germany. [1]

1.6 Thesis Organization

The thesis is organized to provide a clear understanding of the overall work. It consists of six chapters, each of which details one of the several tasks performed to complete it as follows:

Chapter 1: This chapter establishes the Contextual Background for the thesis, beginning with a briefing on the contextual backdrop of LiDAR and SLAM, as well as the GNSS/ IMU systems that facilitate their cooperation. The second section addresses the gaps in previous studies with both Hardware integration/configuration and software availability, followed by the challenges provided by working in an outdoor rather than an indoor environment. The third section describes the thesis primary goals and clear objectives for achieving them. The fourth section then discusses the suggested methodology of how the project will be completed for the hardware and software part. In the final part the study areas for both indoor and outdoor tests are annotated.

Chapter 2: The first section this chapter examines the related works that have been read in conjunction with the currently available Mobile LiDAR solutions. The second section offers a comprehensive analysis of the present state of the art LiDAR technology based on ongoing research and development efforts. The third segment discusses SLAM, including SLAM types and iterative closest point methods. The UTM coordinate system, the Local coordinate system, and the LiDAR coordinate system are annotated in the fourth section. In the final section, the data produced by the Mobile LiDAR RTK (MLRTK) system, including LiDAR packets and NMEA messages, are demonstrated.

Chapter 3: This chapter covers the MLRTK device development. Beginning with the sensors utilized in the creation of the MLRTK, this section provides an overview of the main hardware options and facilities. The subsequent section displays the hardware configuration for the various sensors utilized. In the final section, hardware integration between the sensors is described.

Chapter 4: This chapter starts by discussing the relevant software utilized in this study and why it was chosen. The second part will cover the purpose of the software, the libraries incorporated into the LiDAR RTK software and the LiDAR RTK software functions.

Chapter 5: This chapter begins with the evaluation equipment and software results for the shape of the point cloud in the developed LiDAR RTK and Velodyne Velview. The second section then discusses the accuracy assessment of both indoor and outdoor RTK LiDAR test results. The concluding section then discusses the evaluation criteria used to assess the accuracy of the outdoor and interior test regions.

Chapter 6: This chapter conclude the research findings based on the process of implementation and the obtained results. In addition, the study's limitations and potential recommendations for further research are highlighted.

After the core of the text are the references and annexes.

2 LITERATURE REVIEW

2.1 Related Works

It is essential to go through the previous research in the field of LiDAR integration as well as the use of Mobile LiDAR for surveying applications. There is a shortage of studies that describe the development of LiDAR integration in detail due to the use of such systems by commercial companies. Most researchers present the results of the integrated sensor (i.e., [2], [3],[4]) rather than explaining the hardware integration. Yet, in this study, hardware integration, the performance of the developed Mobile RTK LiDAR along with a comparison with other laser scanners will be taken into consideration.

Author [2] an in-depth performance evaluation of the Velodyne VLP-16 system was conducted to characterize its performance. The study was designed to evaluate the accuracy of VLP16 range estimates as a function of distance and angle of incidence, angular separation between individual beams, and data density as a function of mounting orientation and scanner settings. In addition to testing these key parameters in a well-controlled laboratory, several experiments in the field under realistic conditions were carried out to determine their uncertainties. The study's findings revealed that when used for mapping marine surface features, a low-cost industrial-grade mobile laser scanner can be a cost-effective tool that meets charting requirements. The Velodyne VLP-16 can now be used to validate the vertical clearance of bridges and overhead power cables crossing navigation channels, according to the findings of this study.

In author [3] a novel backpack version of Mobile Laser Scanning (MLS) equipment was introduced for natural science surveying applications requiring precision and mobility in variable terrain conditions. The proposed systems' performance, as determined by analyses of results obtained on a permanent test field and in situ target field studies, shows that the presented MLS systems can generate dense point cloud data for object reconstruction with absolute accuracy of a few centimetres in both plane and elevation.

In [5] by combining multi-sensors such as LiDAR, IMU, GNSS, and a panoramic camera, a portable backpack mobile mapping system was created. The 3D laser SLAM algorithm is used in mobile mapping to acquire geographic information data in a variety of difficult environments. The experimental results in typical indoor and outdoor scenes demonstrate that the system can acquire 3D information with high precision and efficiency, and the relative precision of the point cloud is 24 cm, which meets the requirements of scene mapping and reconstruction.

In author [6] hand-held mobile laser scanners (HMLS) (Zeb1, a 0.7 kg hand-held scanner linked to a netbook computer) were used to conduct a survey by walking across a site and continuously mapping around the surveyor. The accuracy of HMLS data was evaluated by comparing survey results from an eroding coastal cliff site with those acquired by a cutting-edge terrestrial laser scanner (TLS) and photo-survey results processed by structure from motion and multi-view stereo (SfM-MVS) algorithms. After processing the acquired observations, it was discovered that HMLS data had a root mean square (RMS) difference to the benchmark TLS data of 20 mm, which was similar to the SfM-MVS survey (18 mm). The HMLS system's efficiency in complex terrain was demonstrated by acquiring topographic data covering 780 m² of salt-marsh gullies in approximately six minutes, with a mean point spacing of 4.4 cm.

The author[7] presents a study of the current performance of a selection of mobile terrestrial laser scanning devices. This study provided an overview of the integrated positioning, scanning, and imaging devices for these systems. The author believes that mobile laser scanning systems can be separated primarily into two groups (mapping and surveying) based on their eventual use, accuracy, range, and resolution needs. In addition, the author believes that the accuracy requirements for map or survey data vary substantially, thus each scanner's specifications must be evaluated to determine the optimal solution for the intended application. For instance, the TOPCON IP-S2 does not match the requirements for a survey/inspection operation, which requires more precise and thorough point data. However, it can be quite useful for urban mapping applications, which require less precision. More information regarding the outcomes of various LiDAR systems can be found in [7].

2.1.1 Available Mobile LiDAR Systems

In scientific study and experimentation, it is frequently required to prove a theory utilizing already tested and approved devices of a comparable design. This method provides a comparison basis and contributes to the validity and dependability of the outcomes. Using similar technologies to prove a hypothesis can provide a more efficient and cost-effective solution in many instances. Therefore, in my case to demonstrate, I have provided a variety of available Mobile LiDAR equipment for cost comparison.

As illustrated in Table 1, the cost of various handheld mobile LiDAR systems ranging from €32,780 to €49,000. The proposed LiDAR RTK system, on the other hand, could cost the user less than €5,000. The purpose of this research is to develop a MLRTK device based on the integration of LiDAR and RTK GNSS receivers for data collection at surveying sites.

Table 1: Price of a handheld mobile LiDAR system offered by different companies.

Handheld mobile LiDAR	Price	Photo	Reference
ZEB Revo RT	$45,000		[8]
ZEB Horizon	$49,000		[9]
Paracosm PX-80	$38,000		[10]
Greenvalley LiGrip	€32,780		[11]

2.2 State of the Art LiDAR Technology

State of the art is a summary of the most recent advancements and developments in the field of LiDAR technology. The term LiDAR refers to Light Detection And Ranging, is a distant detection and ranging technique by sending out pulses of infrared light and measuring the time for the reflected light to return to the receiver. It is also known as "3D scanning" or "laser scanning". The LiDAR sensor can precisely determine the distance to each object based on the time interval between the outgoing laser pulse and the reflected pulse. Each second, LiDAR collects millions of these highly accurate distance measurement points, from which a 3D map of its surroundings can be created. This thorough mapping of the environment provides data on the position, shape, and behaviour of things.

2.2.1 Laser Return Modes

Returns in LiDAR technology refer to the detection of the laser beam reflections that the LiDAR system emits. Three laser return modes are supported by the VLP-16: Strongest, Last, and Dual. Interactively via the sensor's web interface (where the setting is called Return Type) or programmatically via the curl command, a sensor can be configured to handle laser returns in one of these two ways.[12]

2.2.1.1 Single Return Modes: Strongest, Last

A single return or measurement is obtained when a laser pulse hits a solid wall. In this circumstance, the reading is both the strongest and final response as shown in Figure 4.[12]

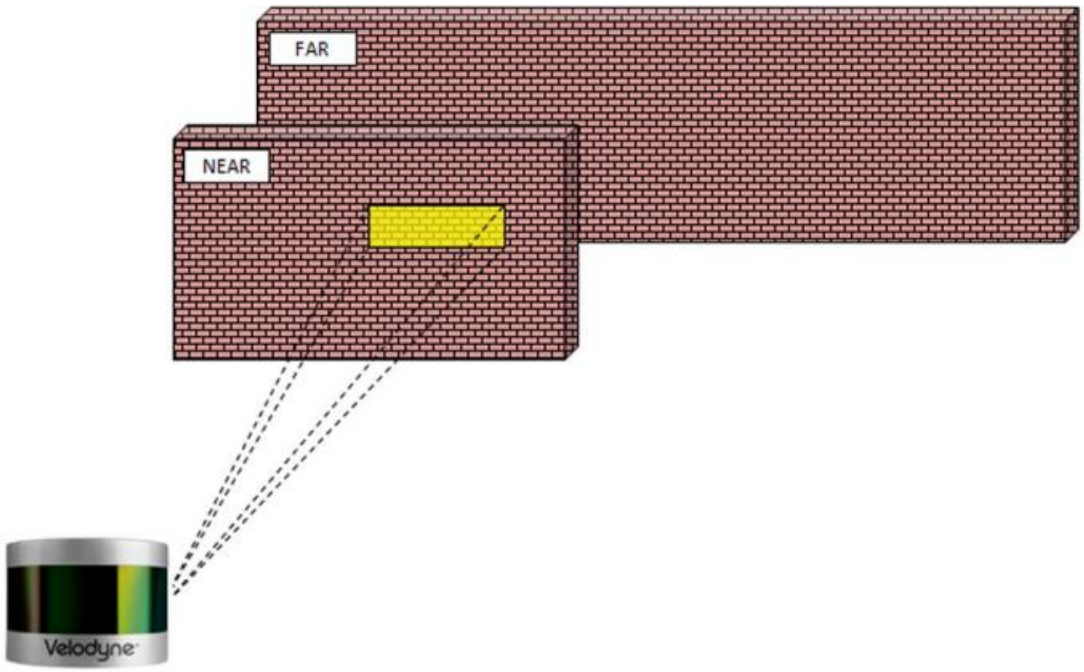


Figure 4: Single Return[12]

2.2.1.2 Multiple Returns

Due to beam divergence, it is possible for a single laser pulse to produce many laser returns. When a laser pulse exits a sensor, it grows slowly and steadily in size. A pulse can be sufficiently large to strike many objects and generate multiple reflections. Typically, the further a reflection originates from, the weaker it is at the detector. Nevertheless, bright or retroreflective surfaces can reverse this effect.[12]

Depending on the laser return mode setting, the VLP-16 analyses multiple returns and reports either the strongest return, the most recent return, or both returns (Figure 5). If the setting is Strongest and a single pulse results in many returns, only the strongest reflection is measured. Similarly, if the setting is Last, only the most recent (in terms of time) reflection will result in a measurement. This could be used to estimate the distance from the air to the ground.

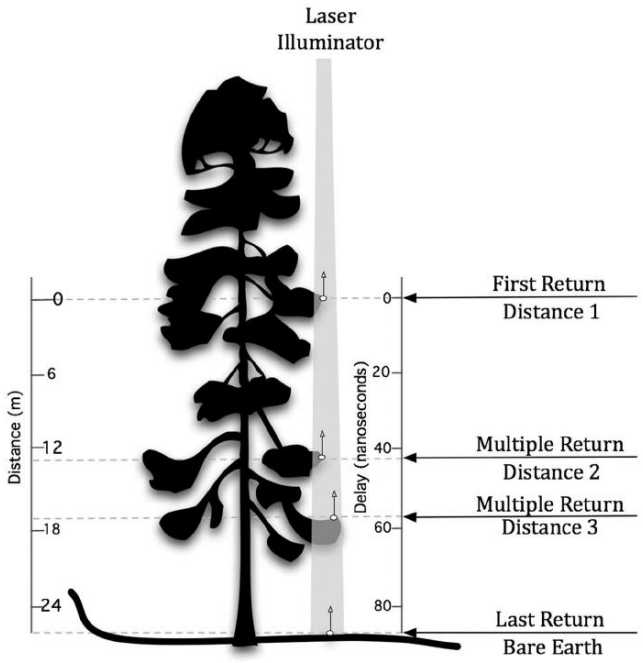


Figure 5: Multiple Returns [13]

2.2.1.3 Dual Return Mode

With dual return LiDAR, the system is intended to detect both the initial and final laser pulse returns. The first return indicates the top of the first object or surface that the laser beam strikes, whereas the last return indicates the last object or surface that the laser beam strikes before returning to the LiDAR receiver. Figure 6 shows the dual return mode provides for the gathering of more precise environmental data; nevertheless, the VLP-16 only captures both returns when the distance between the two objects is at least one meter.[12]

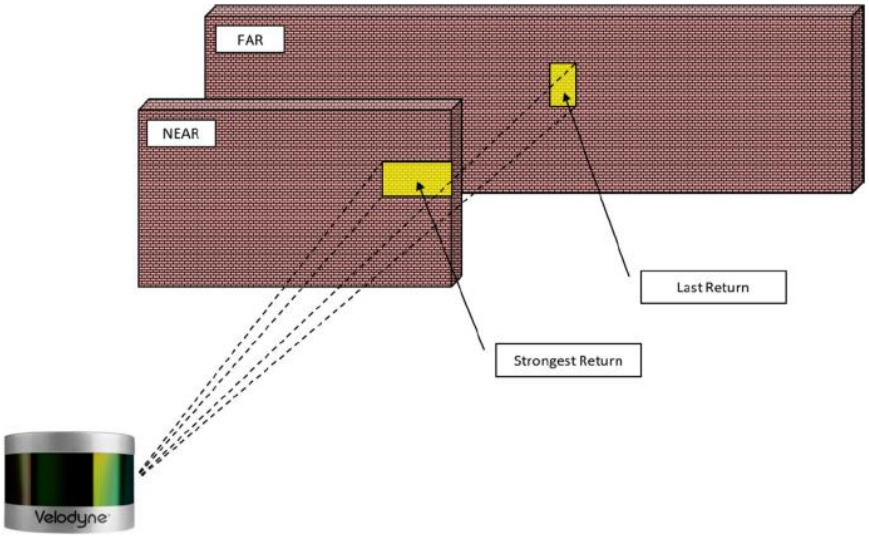


Figure 6: Dual return mode[12]

2.2.2 LiDAR Distance Measuring Methods

LiDAR distance measuring is a technique utilized by LiDAR systems to calculate the distance between a LiDAR sensor and a target. LiDAR operates by producing a laser pulse that travels from the LiDAR sensor to the target and back. The distance to the target is calculated based on the time required for the laser pulse to complete a round-trip.

2.2.2.1 Time of Flight

In LiDAR, the time-of-flight method is the most widely employed method. In this method, the LiDAR system emits a laser pulse and measures the time it takes to return to the receiver. The distance to the target is then determined using the flight time and the speed of light as shown in Figure 7.[14]

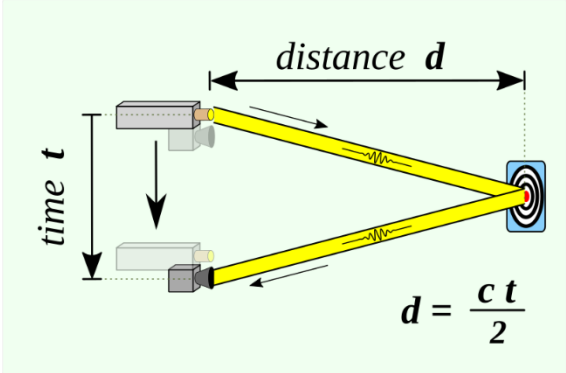


Figure 7: Basic time-of-flight principles applied to laser range-finding[14]

2.2.2.2 Phase Shift Method

With the phase difference method, LiDAR transmits a laser pulse to the target, which is then reflected to the LiDAR receiver. The distance to the target is proportional to the phase difference between the outgoing and arriving laser pulses, which is measured by the LiDAR system.

The phase difference approach is frequently employed in interferometric LiDAR systems, which can provide extremely precise distance readings. In interferometric LiDAR, the entering laser pulse is divided into two independent beams that travel along different routes before being re-combined at the receiver. Using the interference pattern between the two beams, the phase difference between the outgoing and incoming laser pulses is measured.

Figure 8 demonstrates the sensitivity of the phase difference approach to external elements such as vibration, atmospheric conditions, and temperature variations is one of its limitations. In order to provide accurate and trustworthy measurements, LiDAR data must be calibrated and processed with extreme care.

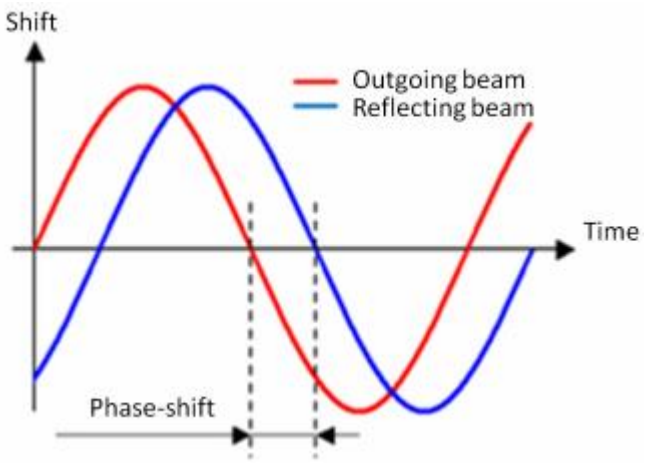


Figure 8:Phase Difference Method to calculate distance [15]

2.2.3 Spinning LiDAR

Spinning LiDAR, also known as spinning or mechanical LiDAR, is a type of LiDAR technology that uses a rotating mirror or prism to rapidly scan a laser beam in many directions and generate a 3D point cloud of a scene. The laser beam is directed in different directions by the rotating mirror or prism, allowing the LiDAR to obtain a 360-degree view of its surroundings.

This is employed because it can identify objects at varied distances from the sensor, and it may be optimal for SLAM. This technology is capable of measuring points in 360 degrees. Some are

meant to measure in a single line (2D scanners), while others are designed to measure in many lines (these referred to as 3D scanner). The Velodyne VLP-32, a high-resolution, 3D laser scanning system that can produce a detailed 360-degree image of the surrounding environment in real-time, is an example of spinning LiDAR. Figure 9 shows the VLP-32 LiDAR that has a range of 200 meters, a 360° horizontal field of view, and a 40° vertical field of view, making it more potent and more expensive than the VLP-16.

Figure 9: Velodyne VLP-32 Spinning LiDAR [16]

2.2.4 Solid-State LiDAR

Solid-state LiDAR sensors are typically available in three different configurations: flash-based, Microelectromechanical System (MEMS)-based, and Optical Phased Array (OPA)-based. Using a photodiode (PD) array, flash-based LiDAR has the benefit of being able to capture the sensing object in a single shot. Since there is no scanning component, flash-based LiDAR has improved long-term dependability and a faster data collecting rate. Key characteristics include a comparison of solid-state and flash-based LiDAR as established in Figure 10. [17]

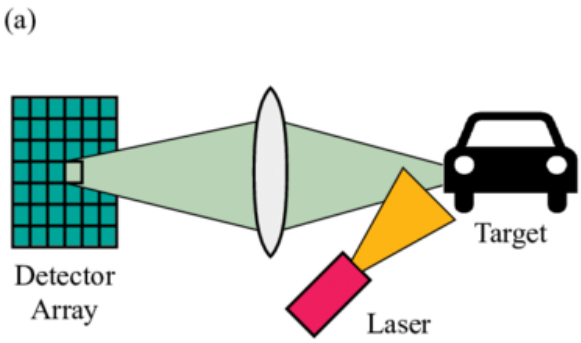


Figure 10: Flash-based LiDAR [18]

On the other hand, MEMS scanning is a technology that controls the direction of light beams in optical communication and imaging systems using micro-scale mechanical parts. A small, movable mirror is connected to a micro-scale mechanical actuator, such as a cantilever beam or comb drive. The actuator can be electrically controlled to alter the mirror's position, so redirecting the light beam. Figure 11 shows MEMS scanning is a popular method for beam steering in laser projectors, barcode scanners, and LiDAR systems, where quick response times and great precision are required. MEMS scanning is superior to existing beam-steering technologies due to its high-speed, high-resolution, and low-cost capabilities, making it an adaptable option for a wide range of applications. [19]

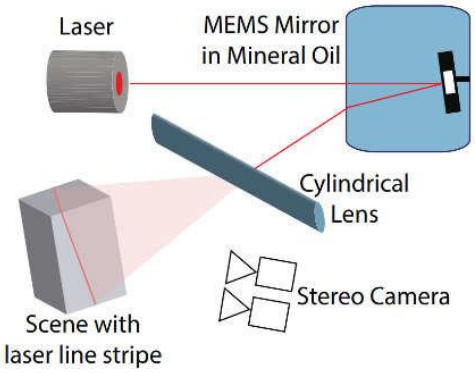


Figure 11: Structured light camera design uses an 1D MEMS mirror and diffused laser [20]

Moreover, Figure 12 shows OPA scanning is a method used in optical communication and imaging systems for regulating and directing light beams. It employs a collection of optical devices, such as tiny lenses or mirrors, that are electronically controlled to alter the phase and direction of light beams in a coordinated manner. By altering the phase and amplitude of the light waves at each element, the system can fast and dynamically steer the light beam in various directions without the need for moving parts. There are numerous possible uses for this technology, including high-speed optical communication, laser range finding, and remote sensing. Due to its high-speed, low-power, and variable beam steering capabilities, it is seen as a promising solution for next-generation optical systems.[19]

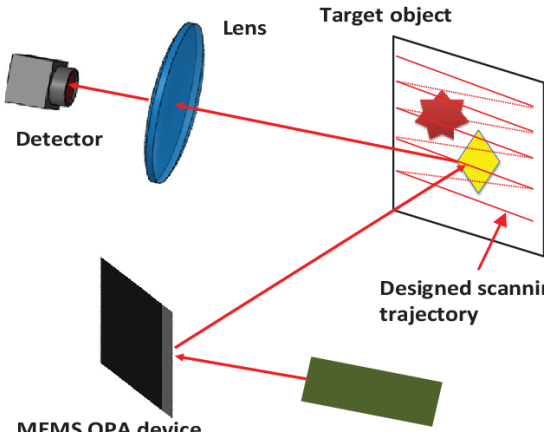


Figure 12: Principle of a typical LiDAR system based on MEMS OPA [21]

One of the solid-state LiDAR is the Velodyne “Velabit” which is used in drones, robots, and autonomous vehicles, the “Velabit” is a small, inexpensive lidar sensor. The “Velabit” has number of benefits, including its tiny size and low power requirements, which make it perfect for use in portable, battery-operated devices. One of the tiniest lidar sensors on the market, the sensor is only 67mm x 70mm x 76.4mm in size and weighs 128 grams as shown in Figure 13.

Figure 13: Velodyne “Velabit” solid state LiDAR [22]

2.2.5 Stationary Mechanic LiDAR Systems

A stationary mechanical LiDAR system is a form of LiDAR system that uses mechanical motion to scan a laser beam across an environment. These systems consist of a stationary platform with a revolving mechanical scanner that directs a laser beam to various scene spots. By monitoring the time, it takes for the laser to return to the detector after bouncing off the target, the system may generate a 3D map of the surrounding area.

Typically, stationary mechanical LiDAR systems are employed for mapping huge regions, such as woods, cities, or industrial sites, and for generating high-resolution 3D representations of objects and sceneries. In robotics and autonomous vehicles, they are frequently utilized for autonomous navigation and obstacle avoidance.

In comparison to other LiDAR systems, such as flash LiDAR and MEMS scanning LiDAR, stationary mechanical LiDAR systems have the benefit of being able to cover a vast field of view and produce high-resolution 3D maps with a single scan. However, they have certain drawbacks, including a relatively sluggish scanning speed and a larger size and weight, which can hinder their mobility and portability.

One of the stationary mechanical LiDAR systems is the Trimble X7 Static LiDAR (Figure 14). It employs a stationary mechanical scanner to build a 3D environment map. The Trimble X7 operates at 500,000 points/second. In addition, it has a measurement range of up to 80 meters and a can scan 360 degrees.

Figure 14: Trimble X7 Static LiDAR [23]

2.3 Simultaneous Localization And Mapping (SLAM)

The computational challenge of creating or updating a map of an uncharted area while tracking an agent's position inside it is known as SLAM. There are various algorithms that can solve it in a particular context in at least somewhat tractable time. The particle filter, extended Kalman filter, covariance intersection, and Graph SLAM are examples of popular approximation techniques. SLAM algorithms, which are based on concepts in computational geometry and computer vision, are used in robotic navigation, robotic mapping, and odometry for virtual reality or augmented reality. Depending on the requirements of the application, many SLAM approaches might be utilized.[24]

2.3.1 Types of SLAMS

Depending on the requirements of the application, many SLAM approaches might be utilized. SLAM is a robotics technology used to simultaneously generate a map of an unknown

environment and determine its placement within it. Several types of SLAM algorithms are compatible with LiDAR sensors, including:[24]

- **EKF-SLAM** (Extended Kalman Filter SLAM): This algorithm uses an extended Kalman filter to estimate the robot's pose and the location of landmarks in the environment.
- **Fast SLAM**: This algorithm uses a particle filter to estimate the robot's pose and the location of landmarks in the environment. It can handle non-linear and non-Gaussian systems and is particularly useful in large-scale mapping.
- **Graph SLAM**: This algorithm uses a graph to represent the environment and the robot's path through it. It estimates the robot's pose and the location of landmarks by minimizing the error between the predicted and observed sensor data.
- **ICP-SLAM** (Iterative Closest Point SLAM): This algorithm uses the iterative closest point algorithm to align successive LiDAR scans and build a map of the environment. It is particularly useful in indoor environments where the structure of the environment is relatively simple.
- **LOAM** (Lidar Odometry and Mapping): This algorithm uses a scan matching technique to estimate the robot's pose and the location of landmarks in the environment. It is particularly useful in outdoor environments where the environment is more complex.

Each of these algorithms has its own strengths and weaknesses, and the choice of algorithm will depend on the specific requirements of the SLAM application. [24]

2.3.2 Iterative Closest Point (ICP)

ICP is a commonly used approach in 3D registration and alignment applications, including point cloud registration and 3D surface matching. There are a variety of ICP algorithms that can be utilized depending on the needs of a given application.[25]

2.3.2.1 Point-to-Point ICP

Minimizes the distance between comparable points in two-point clouds. The algorithm iteratively modifies the relative position of two-point clouds until the distance between matching points is minimized. Point-to-point ICP is generally quick and can tolerate noise in the data, but it may not function effectively if there are substantial geometric variances between the two-point clouds.[25]

2.3.2.2 Point-to-Plane ICP

In Point-to-Plane ICP, the method minimizes the distance between each point in one point cloud and the plane specified by the point in the other point cloud to which it corresponds. This approach is capable of handling more intricate geometric differences between two-point clouds and can produce more precise registration results.[25]

2.3.2.3 Robust ICP

Robust ICP employs a statistical technique to handle outliers and other data noise. The algorithm iteratively modifies the position of one point cloud in relation to another, while simultaneously estimating the probability distribution of the correspondences and weighting them accordingly. This strategy can yield more precise and reliable registration results, particularly in noisy or cluttered surroundings.[26]

2.3.2.4 Non-rigid ICP

ICP that is not rigid permits deformations and shape changes between the two-point clouds. This method can be utilized for applications such as surface registration and form modeling, in which the point clouds may have considerable shape or topology discrepancies. Non-rigid ICP methods are often slower than rigid ICP methods, but they can yield more precise and adaptable registration results.[27]

2.4 Coordinate Systems

It is crucial to give a brief description of the various coordinate systems that were applied in this study. For analysis purposes, it was also required to convert the various coordinate systems to a uniform one. To evaluate and interpret the data using a single coordinate system, for instance, GPS coordinates were converted to UTM coordinates.

2.4.1 Universal Transverse Mercator (UTM)

The UTM is a map projection system for assigning coordinates to locations on the surface of the Earth which can be seen in Figure 15. Like the traditional method of latitude and longitude, it is a horizontal position representation, which means it ignores altitude and treats the earth as a perfect ellipsoid. However, it differs from global latitude/longitude in that it divides earth into 60 zones and projects each to the plane as a basis for its coordinates. Specifying a location means

specifying the Zone and the East and North coordinate in that plane. The projection from spheroid to a UTM zone is some parameterization of the transverse mercator projection. The parameters vary by nation or region or mapping system[28]. Most zones in UTM span 6 degrees of longitude, and each has a designated central meridian. The scale factor at the central meridian is specified to be 0.9996 of true scale for most UTM systems in use[28]. The GPS coordinates (latitude and longitude) of Palestine area were converted to a UTM East and North. It is worth mentioning that the Palestine test area is located within the 36S UTM zone.

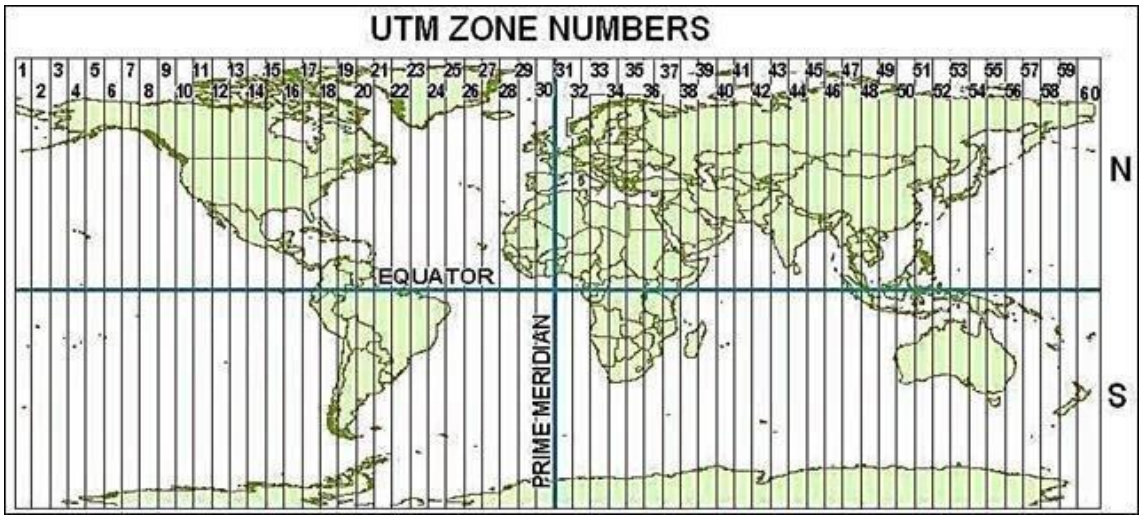


Figure 15: Universal Transverse Mercator[29]

2.4.2 Local Coordinates System

A local coordinate system is a set of coordinates specified at a particular location or region in space. The local coordinate system is used to describe the location, orientation, and motion of objects with respect to a specific point or region [30]. In a local coordinate system, the directions of the x, y, and z axes relative to the local point or region are represented by a collection of basis vectors as shown in Figure 16. The point or region of interest is normally the origin of the local coordinate system, and the basis vectors are often orthogonal (perpendicular)[30].

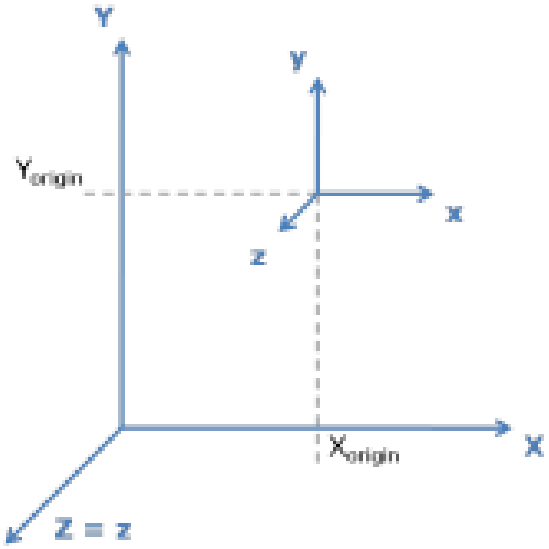


Figure 16: 3D Local Coordinate system [31]

2.4.3 LiDAR Coordinate System

LiDAR data typically employ a 3D Cartesian Local coordinate system, where the X, Y, and Z axes represent the horizontal, lateral, and vertical dimensions, respectively. The location of the LiDAR scanner, which is often installed on a moving platform such as an aircraft, automobile, or portable device, defines the origin of the coordinate system. The orientation of the coordinate system is influenced by the orientation of the LiDAR scanner and the direction of gravity.[32]

In conclusion, the center of each frame created by LiDAR contains the frame's coordinate (0, 0, 0). Also, when matching frames using ICP as described in section 2.3.2, the SLAM system retrieves the pose by matching the current frame to the reference frame (s). The reference frame(s) may be the previous frame, the previous few frames, or the map thus far generated as illustrated in Figure 17. To do this matching operation, the system must be able to identify feature points that correlate between the current frame and the reference frame (s). In general, the system defines as "correspondence" a feature point in the reference nearest to a feature point in the present frame. [32]

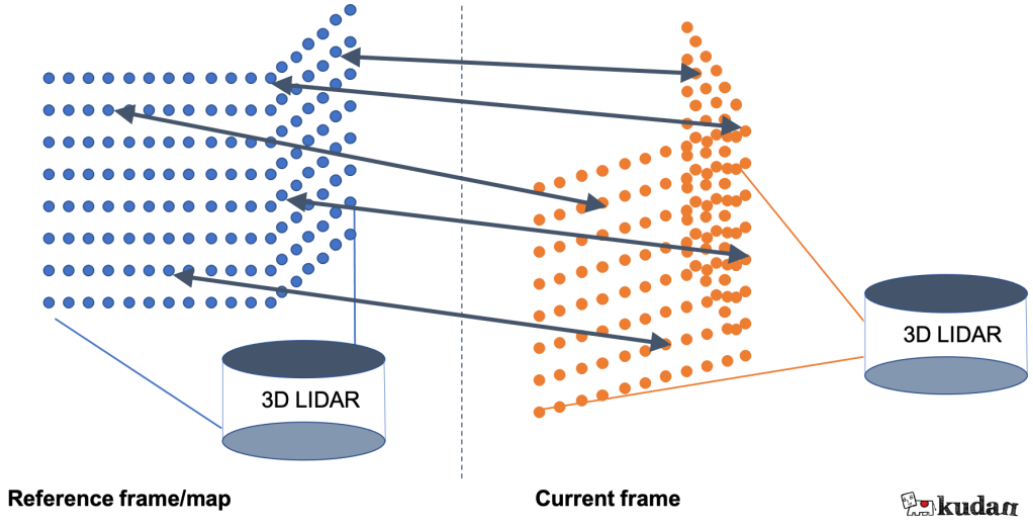


Figure 17: Finding correspondences between the current frame and the reference frame [32]

The methodology is a crucial component of the study since it describes the processes performed to collect and analyze data. This can comprise the research design, sample selection, methods of data collecting, and data processing procedures.

2.5 Coordinates Transformation

In surveying and mapping, the transformation of points from one coordinate system to another is a typical issue. In addition, coordinate transformations are widely utilized in photogrammetry and GPS surveying to convert the measured coordinates to the required coordinate system. Transformation can be conducted on 2D coordinates or on 3D coordinates as required. The following section will provide a brief discussion about the common transformation used in surveying.

2.5.1 A Two-Dimensional Transformation

The three primary 2D transformation methods are: the conformal transformation, the affine transformation, and the polynomial transformation. The following section will focus on the 2D conformal transformation since it was used in the development of the LiDAR RTK software.

- **Conformal transformation**: The two-dimensional conformal coordinate transformation, also known as the four-parameter similarity transformation, has the characteristic that true shape is preserved after transformation. It is typically used in surveying when converting separate surveys into a common reference coordinate system. This transformation is a three-step process that involves:

1. **Scaling**: A scaling transformation stretches or compresses distances in the plane by a fixed factor. It creates equal dimensions in the two coordinate systems. To make line lengths as defined by the (*x, y*) coordinate system equal to their lengths in second system, it is necessary to multiply x, y coordinates by a scale factor, *S*. Thus, the scaled coordinates *x'* and *y'* are:

$$x' = S \times x \ , \quad y' = S \times y \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 1$$

2. **Rotation**: A rotation transformation rotates every point in the plane by a fixed angle around a fixed point. It makes the reference axes of the two systems parallel. Expressions that give the (*X', Y'*) rotated coordinates by rotation angle (*θ*) for any point in terms of its (*x', y'*) coordinates are:

$$X' = x' \cos\theta - y' \sin\theta \ , \quad Y' = x' \sin\theta + y' \cos\theta \quad \ldots\ldots\ldots\ldots\ldots 2$$

3. **Translation**: A translation is a transformation that moves every point in the plane by a fixed distance in a fixed direction. It creates a common origin for the two coordinate systems. To finally arrive at *X* and *Y* coordinates for a point, it is necessary to translate the origin of the *X'* and *Y'* system to the origin of the *X* and *Y* system by translation values (*Tx* and *Ty*)as follows:

$$X = X' + Tx \ , \quad Y = Y' + Ty \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3$$

If Equations (1), (2), and (3) are combined, a single set of equations results that transform the points of first coordinate system (*x* and *y*) directly into second coordinates system (*X* and *Y*) as follows:

$$X = (S\cos\theta)x - (S\sin\theta)y + Tx \ , \quad Y = (S\sin\theta)x + (S\cos\theta)y + Ty \quad \ldots\ldots 4$$

- **Affine transformation**: It is a linear (or first-order) transformation which applies coordinate transformation through a rotation, a scale change in x- and y-direction, followed by a translation. The affine transformation applies scaling factor on both X and Y directions, while the conformal transformation applies one overall scale factor. The affine transformation function is expressed with 5 parameters: one rotation angle (*θ*), two scale factors, a scale factor in the x-direction (*Sx*) and a scale factor in the y-direction (*Sy*), and two origin shifts (*Tx*, *Ty*). The equation is:

$$X = (Sx\cos\theta)x - (Sy\sin\theta)y + Tx \ , \quad Y = (Sx\sin\theta)x + (Sy\cos\theta)y + Ty \quad \ldots\ldots 5$$

- **Polynomial transformation**: It is a non-linear transformation which applies coordinate transformation through a translation, a rotation, and a variable scale change. The transformation function can have an infinite number of terms. The equation is:

$$X = x_o + a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2 + a_6x^2y + a_7xy^2 + a_8x^3 + \cdots$$
$$Y = y_o + b_1x + b_2y + b_3xy + b_4x^2 + b_5y^2 + b_6x^2y + b_7xy^2 + b_8x^3 + \cdots 6$$

2.5.2 A Three-Dimensional Transformation

The 3D transformation applies rotation, scaling, and translation to the 3D (X, Y, and Z) coordinates, much like the 2D transformation does. The two most applied methods for transformation of 3-dimensional coordinates are:

- **Geocentric translation**: It is a three-parameter translation between two sets of geocentric coordinates. This shift is defined by the parameters *dx*, *dy* and *dz*.
- **Helmert 7-parameter transformations**: Three-dimensional Helmert transformation is a similarity transformation that defined by seven parameters and is therefore also called 7-parameter transformation. The seven parameters are as follows: 3 translations along each axis: *Tx*, *Ty*, and *Tz*; 3 rotations: one around each coordinate axis: *Rx*, *Ry*, and *Rz*; and a scale change, commonly expressed as a scale correction in ppm. It is commonly used in geodesy to produce datum transformations between datums.

2.6 Interpolation

In surveying calculations, interpolation is commonly used for estimating the coordinates of a point between two known points. The method of interpolation is used within the LiDAR RTK software for interpolating x, y, and z data between two known points and to create a smooth trajectory. This section will provide an overview about the major types of interpolation as shown in Figure 18.

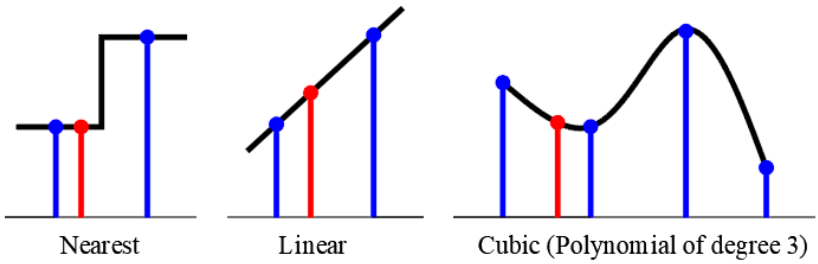


Figure 18: Types of interpolation

- **Nearest Neighbor Interpolation**: This method applies the closest known value to calculate the value between two known values. It is useful when there are limited data points available.
- **Linear Interpolation**: This method requires a straight line between two points. In this research, it is used within LiDAR RTK software for performing distance-based

interpolation and time-based interpolation. Distance-based linear interpolation is a method of estimating a value between two known values based on their respective distances or positions along a line. It involves finding the value that lies between two data points at a specific distance by calculating a weighted average of the two points based on their distance from the point being estimated. As well, time-based linear interpolation is like distance-based interpolation, however, the timestamp used instead of distance.

- **Polynomial**: Polynomial interpolation is a method of estimating a polynomial function that passes through a set of data points. It involves finding a polynomial of degree "n" that goes through "n+1" data points. The polynomial can then be used to estimate values between the data points or to extrapolate values outside the range of the data points. One of the more common methods of polynomial interpolation is cubic spline interpolation, which is described by four coefficients and referred to as a third-degree polynomial. The general formula of a polynomial of degree n is illustrated in equation 7.

$$f(x) = a_o + a_1 x + a_2 x^2 + \ldots + a_n x^n \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 7$$

2.7 RTK NMEA Messages

NMEA stands for National Marine Electronics Association and it is a protocol for transmitting information about marine electronics, including GPS receivers, using ASCII text messages sent over a serial communication link. A sample of NMEA file that includes information about different NMEA messages such as GPRMC (GPS Recommended Minimum Specific GPS/TRANSIT Data), GPVTG, GPGGA, and GPGSV are shown in Figure 19.

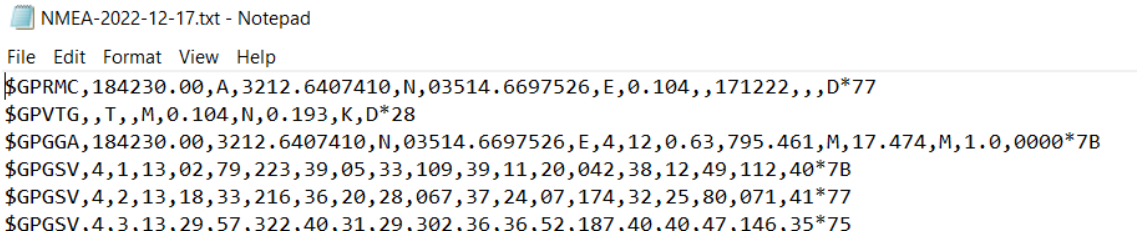


Figure 19: NMEA message sample

Mainly there are two NMEA messages that were utilized in this research which are GPRMC and GPGGA messages. The GPRMC is a phrase structure used in the NMEA 0183 standard for transferring GPS data. The GPRMC message consists of a series of comma-separated fields that contain specific pieces of information. For example, the first field in the GPRMC message is the time stamp, which indicates the time that the GPS data was recorded. The second field is the status

indicator, which indicates whether the GPS receiver is active or inactive. The third and fourth fields contain the latitude and longitude coordinates, respectively, and the fifth field contains the speed over ground. The sixth field contains the course over ground, and the seventh field contains the date. The eighth field includes Magnetic variation, in degrees, and the ninth field has the checksum data. The GPRMC sentence gives critical GPS data, as shown in Figure 20 and demonstrated in Table 2:

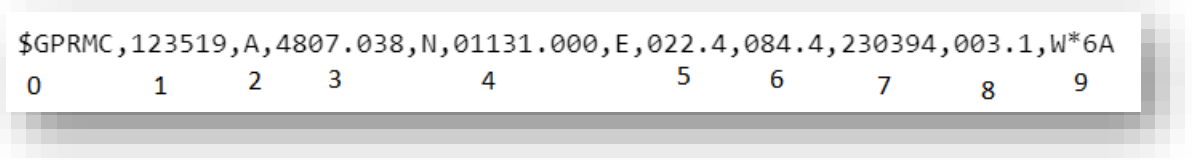


Figure 20: GPRMC message example

Table 2: GPRMC Message[33]

Field	Meaning
0	Message ID $--RMC Talker ID can be: GP: GPS only GN: More than one constellation
1	UTC of position fix
2	Status A=active or V=void
3	Latitude
4	Longitude
5	Speed over the ground in knots
6	Track angle in degrees (True)
7	Date
8	Magnetic variation, in degrees
9	The checksum data, always begins with *

GPGGA Message: Global Positioning System Fixed Data (GPGGA) is a specific sort of NMEA message. It is used to transmit position data, including latitude, longitude, and altitude, from a

GPS receiver. GPGGA message is used to convey the LiDAR system's GPS-determined position. The message would include fields for the latitude, longitude, and altitude of the LiDAR, as well as additional information such as the number of satellites utilized to establish the position, the position's precision, and the measurement time. Figure 21 displays the GPGGA which will be parsed to show each field and its meaning and demonstrated in Table 3:

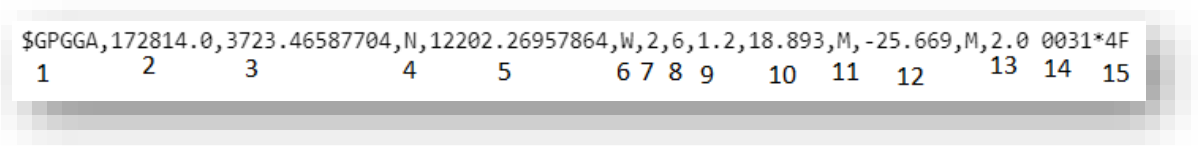


Figure 21: GPGGA Message example

Table 3: GPGGA Message [34]

Field	Meaning
0	Message ID $GPGGA
1	UTC of position fix
2	Latitude
3	Direction of latitude: N: North S: South
4	Longitude
5	Direction of longitude: E: East W: West
6	GPS Quality indicator: 0: Fix not valid 1: GPS fix 2: Differential GPS fix (DGNSS), SBAS, OmniSTAR VBS, Beacon, RTX in GVBS mode 3: Not applicable 4: RTK Fixed, xFill

	5: RTK Float, OmniSTAR XP/HP, Location RTK, RTX 6: INS Dead reckoning
7	Number of SVs in use, range from 00 through to 24+
8	HDOP
9	Orthometric height (MSL reference)
10	M: unit of measure for orthometric height is meters
11	Geoid separation
12	M: geoid separation measured in meters
13	Age of differential GPS data record, Type 1 or Type 9. Null field when DGPS is not used.
14	Reference station ID, range 0000 to 4095. A null field when any reference station ID is selected, and no corrections are received.
15	The checksum data, always begins with *

2.8 LiDAR PCAP data

PCAP File: PCAP files include LiDAR point cloud data delivered through a network connection, such as Ethernet or Wi-Fi. The PCAP file can be used to collect and analyze LiDAR data as it is transmitted, which is useful for debugging and enhancing the LiDAR system's performance.

The VLP-16 LiDAR sends data through its Ethernet port as User Datagram Protocol (UDP) packets [35]. In addition, it generates two types of packets: Data packets and Position packets. Position packets are also known as GPS packets and telemetry packets. [12]

Data packets include both the measured 3D data and the calibrated reflectivity of the surface from which the light pulse was reflected. The data packet also contains a series of azimuths, a 4-byte timestamp, and two factory bytes indicating the sensor model and laser return mode. Understanding the model and return mode gives your software the knowledge it needs to automatically adapt to various data formats.[12]

If you've configured your sensor to synchronize with a GPS time source, position packets give a copy of the last NMEA message received (either GPRMC or GPGGA). Position packets also contain a byte indicating the PPS signal's state for synchronization with a time source. Upgraded firmware gives further data.[12]

2.8.1.1 Data Packet Structure

A data packet of 1248 bytes is transmitted through UDP on port 2368. The data packet has a protocol header of 42 bytes, 12 Data Blocks, a 4-byte timestamp, and 2 factory bytes.

There are two types of data packet formats:

- Single Return Mode
- Dual Return Mode

Figure 22 demonstrates the Single Return Mode packet data structure. The Dual Return Mode packet data structure is depicted in Figure 23. There are number of significant distinctions in the packet structures. In Dual Return Mode, the sensor first transmits a pair of data blocks for each azimuth firing angle. The odd-numbered blocks (1, 3,..., 9, 11) include the strongest or second-strongest return, whereas the even-numbered blocks (0, 2,..., 8, 10) contain the weakest or second-weakest return. If the highest return is also the final return, the second-highest return is offered. If only one return was identified, the even|odd block pairings (0|1, 2|3, 4|5, 6|7, 8|9, 10|11) will contain identical data.[12]

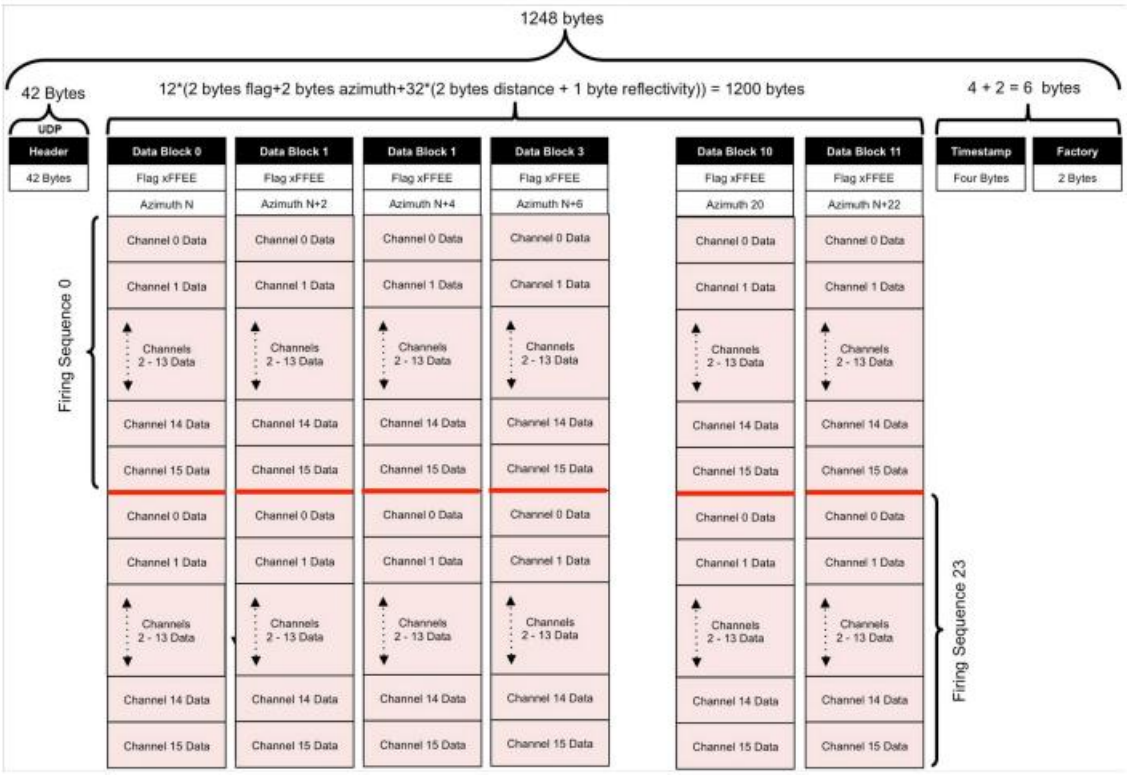


Figure 22: VLP-16 single return mode data structure[12]

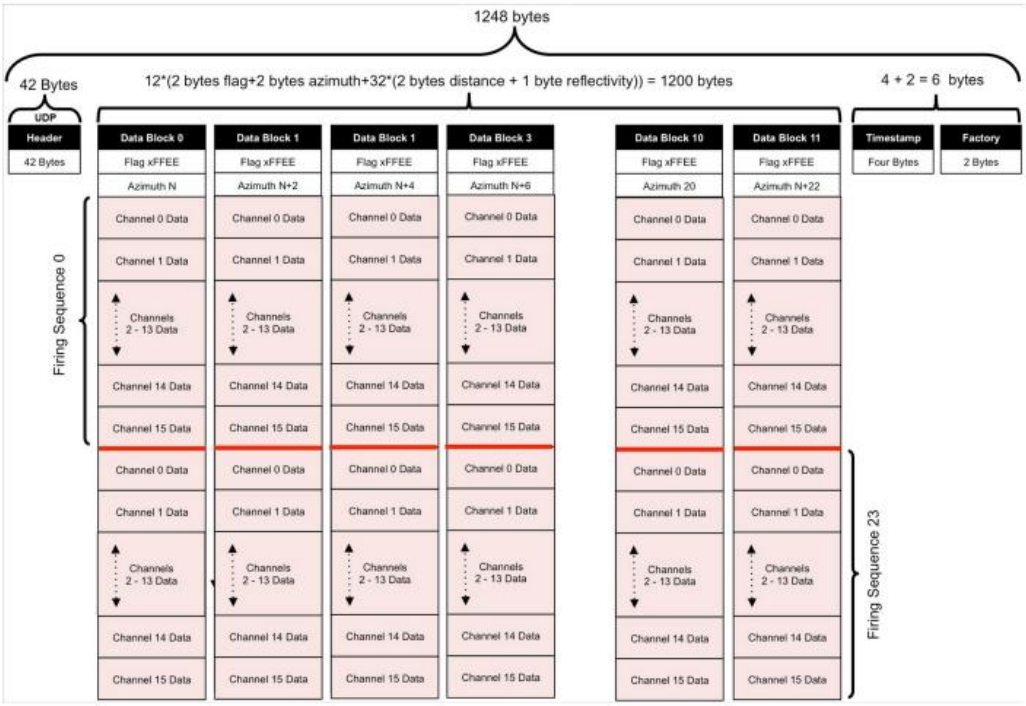


Figure 23: VLP-16 dual return mode data structure [12]

2.8.1.2 Position Packet Structure

The Position Packet is responsible for providing a copy of the most recent supported NMEA phrase received from an external GPS/INS/IMU source, as well as the Pulse Per Second status, a time stamp indicating when the position packet was generated, and optionally other relevant information. If there is no GPS/INS/IMU attached or if it is disabled, the NMEA phrase, PPS status, and associated fields in the position packet will be empty (i.e., all zeros). The position packet is a UDP packet of 554 bytes received on port 8308 (by default). Protocol headers comprise the initial 42 bytes. Length of the payload is 512 bytes. Figure 24 demonstrates the position packet to a GPRMC message using Wireshark software, the GPRMC message has been illustrated in section 4.3.1.[12]

```
00d0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00f0  0f 36 0d 98 02 00 00 97  24 47 50 52 4d 43 2c 31   .6...... $GPRMC,1
0100  38 34 32 33 30 2e 30 30  2c 41 2c 33 32 31 32 2e  84230.00 ,A,3212.
0110  36 34 30 37 34 31 30 2c  4e 2c 30 33 35 31 34 2e  6407410, N,03514.
0120  36 36 39 37 35 32 36 2c  45 2c 30 2e 31 30 34 2c  6697526, E,0.104,
0130  2c 31 37 31 32 32 32 2c  2c 2c 44 2a 37 37 0d 0a  ,171222, ,,D*77..
0140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

Figure 24: Wireshark Position Packet

3 Developing a Mobile RTK LiDAR Device

Implementation refers to the design and construction of a model based on hardware and software needs and standards. It includes translating abstract concepts, algorithms, and designs into physical components that can execute functions.

3.1 Hardware Sensors

Related hardware components provide an overview of the available and relevant hardware options and facilitates a comparison of the capabilities and performance of the various options. These components collaborate to carry out the required activity of the mobile LiDAR system. This section outlines the sensors utilized in the development of the MLRTK system.

To build a mobile LiDAR system, several devices are utilized as follows:

- **Velodyne VLP-16 LiDAR**: This LiDAR in Figure 25 in the VLP-16 which has 16 channels and can generate 300,000 points per second from a 360° horizontal field of view and a 30° vertical field of view with a ±15° elevation from the horizon. The cost of the VLP-16 for educational use costs about 2800€, however, for commercial use it costs about 3200€.

Figure 25: Velodyne VLP-16

Time-of-flight (ToF) approach is used by VLP-16 LiDAR sensors. Each Infra-Red laser's time of shooting and direction are recorded when it fires a laser pulse. The laser pulse continues through the air until it encounters a barrier, where some of the energy is reflected. The paired IR detector takes in some of that energy and records the time-of-acquisition and power received.

LiDAR can emit MILLIONS of laser points per second, for VLP-16 it can emit 300,000 points per second. Typically, the number of beams emitted by a 3D LiDAR is referred to as the "number of channels", 16 for VLP-16. Each laser scan is composed of multiple pulses, each pulse is composed of three short closely-spaced horizontal bars or bands of light. The gap between scanlines can be calculated with the following equation (7):

$$Gap = distance\ to\ target * \tan(vertical\ angle\ between\ scan\ lines) \quad \ldots\ldots\ldots\ldots 8$$

VLP-16 Interface Box, the Interface Box provides convenient power, Ethernet, and GPS input connections. It safeguards the sensor from power fluctuations by including a replacement fuse and a diode for reverse-current protection. Moreover, the VLP-16 can be used without an Interface Box, but the user must provide appropriate reverse- and over-voltage protection circuitry to safeguard the sensor if used without an Interface Box.

Each data block starts with the flag bytes followed by a two-byte azimuth value (α). It is an unsigned integer, the azimuth. It symbolizes a fractional degree angle. Azimuth can have values between 0 and 35999. Per data block, just one azimuth value is given.

The first data point in the first firing sequence of the first data block is marked by the four-byte time stamp, which is a 32-bit unsigned integer. The value of the time stamp represents the number of microseconds that have passed since the hour's top. The value represents the number of microseconds in an hour and varies from 0 to 3,599,999,999. Because geo-referencing software uses the time stamp to link each laser firing with related information from an inertial navigation system, it is essential. A sequence of time-stamped values for pitch, roll, yaw, latitude, longitude, and elevation are provided by the inertial navigation system. The user's program can mathematically transfer the data from the sensor's coordinate frame to an earth-based reference frame by comparing the time of the data point to the time-stamped data from the INS. The GPS/Universal INS's Coordinated Time (UTC) is used to match the time stamps.

The sensor starts counting microseconds as soon as it is powered on using an internal time reference. However, you can determine the precise firing time of each laser in any given packet since the sensor can synchronize its data with UTC time.

A user-supplied GPS/INS receiver that produces a synchronizing Pulse Per Second (PPS) signal and an NMEA GPRMC message is required for UTC synchronization. The minutes and seconds are given in UTC in the GPRMC message. After synchronization, the sensor

reads the minutes and seconds from the GPRMC message in order to set the sensor's time stamp to the precise number of microseconds past the hour according to UTC. Moreover, Figure 26 shows how the LiDAR sweeps the surrounding environment.

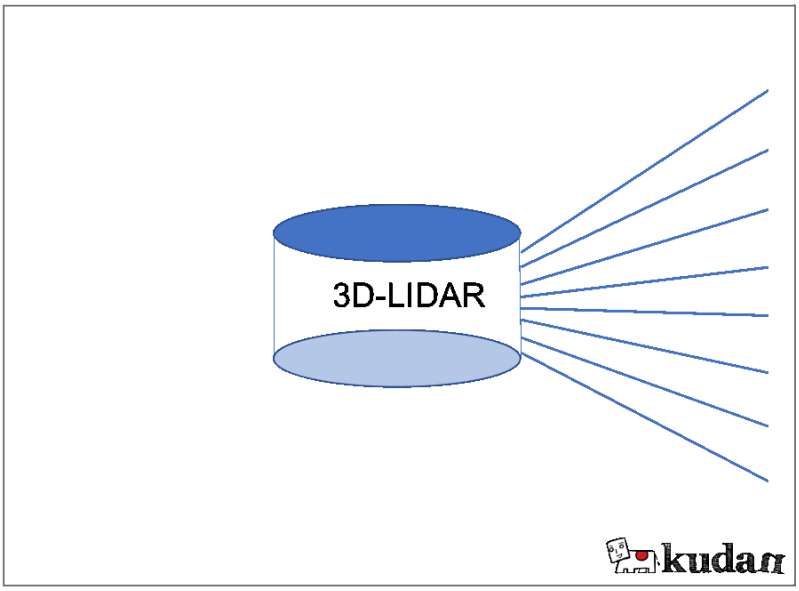


Figure 26: How a LiDAR sweeps the surrounding environment [36]

- **SimpleRTK2B GNSS module**: A multi-band GNSS receiver based on the u-blox ZED-F9P is simpleRTK2B a budget board (module) was bought for about 230€ to be used in this project which can be seen in Figure 32. The main component of simpleRTK2B is u-blox ZED-F9P. The simpleRTK2B board can be powered from 4 different sources: (GPS USB port, XBEE USB port, Pixhawk connector, Arduino rail) as shown in Figure 27.

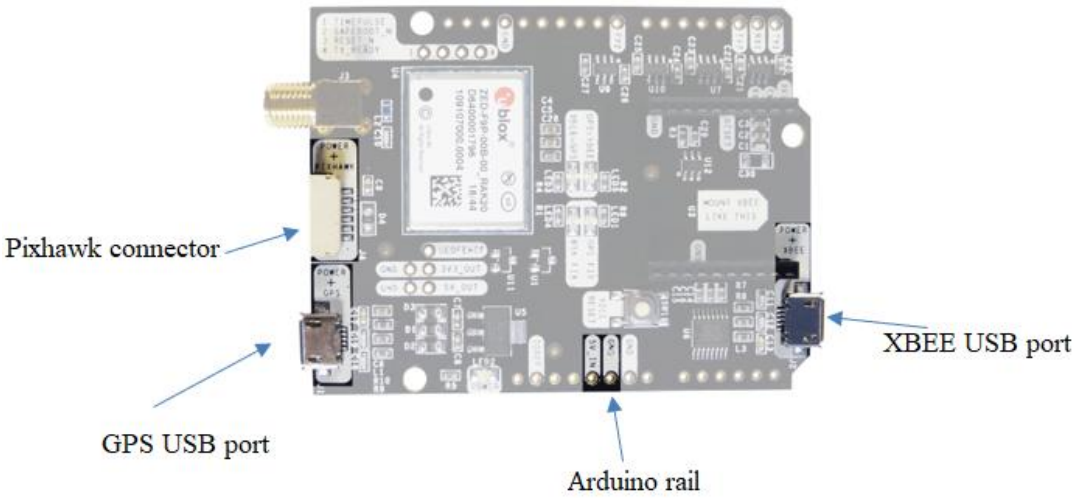


Figure 27:simpleRTK2B power sources

Additionally, these ports are utilized for communication through which NMEA message and position data can be received. Additionally, using the Arduino rails, it is possible to connect an Arduino. The SimpleRTK2B has Centimeter level precision (<1cm with a base station up to 35km and <1cm with NTRIP up to 35km). Also, it has Update rate default (1Hz, with maximum performance: up to 10Hz and with reduced performance: up to 20Hz). Moreover, it has Multifrequency for the following bands (GPS: L1, L2), (GLONASS: G1, G2), (BeiDou: B1, B2), (Galileo: E1, E5b), (QZSS: L1, L2), (SBAS: WAAS, EGNOS, MSAS and GAGAN). Accordingly, it takes the GPS 25 seconds (cold), 2 seconds (hot start) to get the first fix position, and 35 seconds to start up and obtain the initial fix RTK location (cold start).

To conduct an accurate RTK survey, a rover and a base that communicates adjustments in real time are typically required. However, you can choose between using your own base or a remote one using NTRIP via Internet Protocol. NTRIP enables your rover to receive corrections over the Internet without requiring a second nearby receiver to serve as a base. NTRIP consists of three major components: the base, the caster, and the client's rover. Typically, a stationary continuously operating reference station, or CORS, is present in this scenario. The data is subsequently delivered to the NTRIP caster, which re-transmits it via the Internet port to the approved client rover connected via a specific port. NTRIP is a very useful method to implement when working in RTK. However, it has its advantages and disadvantages. Using NTRIP you can obtain High absolute accuracy, and there will be no need for a local base to get the essentially corrections and you will be able to perform fast fix solution. Nevertheless, the NTRIP correction are dependent on internet thus, cellular coverage is essential. Likewise, some NTRIP networks are private and require payments which can be sometimes quite expensive, while others are free of charge.

There is a considerable probability that you will encounter this word if you work with NTRIP. Virtual Reference Station, or VRS, is a valuable tool for lengthy baselines.

The concentration of NTRIP bases and CORS might vary substantially from location to location. If the station density is sufficient, but you are too far from any of them, VRS allows you to erase your baseline.

Consider that you are 70 kilometers from the nearest station. In RTK mode, this baseline is fairly extensive. Your rover sends data back to the NTRIP cast to generate a VRS. NTRIP permits the merging of data from your rover and dense NTRIP bases, simulating

a virtual base next to you. Your baseline will therefore decrease from 70 kilometers to 0 km.

A base station is a fixed location where the position and timing of GPS signals are accurately known. Typically, base stations are used to deliver precise position and time information to rovers, which can subsequently use this information to precisely establish their own position. This method is referred to as differential GPS or DGPS. To use differential GPS, the base station must be equipped with a GPS receiver that can determine its own position and timing with high precision. The base station then transmits this data to the rover along with any GPS signal modifications required to account for mistakes and biases. This data is utilized by the rover to estimate its position with a high degree of precision.

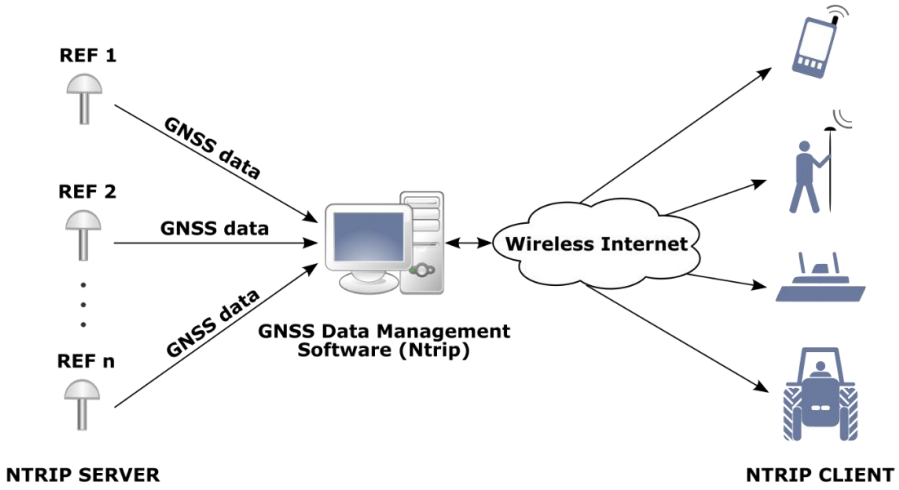
There are several advantages to using base stations in Global Positioning System (GPS) applications:

1. By using a base station to deliver accurate position and timing information to a rover, differential GPS (DGPS) can attain significantly more precision than independent GPS. This is since the base station can rectify faults and biases in the GPS signals that would otherwise impact the rover's position accuracy.
2. Coverage: Base stations can give coverage in regions where GPS signals may be weak or absent, such as urban canyons or dense vegetation.
3. Base stations can be used to offer a dependable source of position and timing information when GPS signals are absent or disturbed, such as during a satellite outage or interference.

While there are many advantages to using base stations in GPS applications, there are also some potential disadvantages to consider:

1. Installation and maintenance of a base station can be costly, especially if numerous base stations are required to cover a broad area.
2. Infrastructure: Base stations need a dependable supply of electricity and a secure position with a clear view of the sky. In some instances, it may be impossible or impractical to locate a base station in a suitable position.
3. Base stations require routine maintenance to maintain their proper operation and the accuracy of the position and timing information they supply. This can be time-intensive and expensive.

Figure 28 shows the difference between NTRIP Caster and VRS.



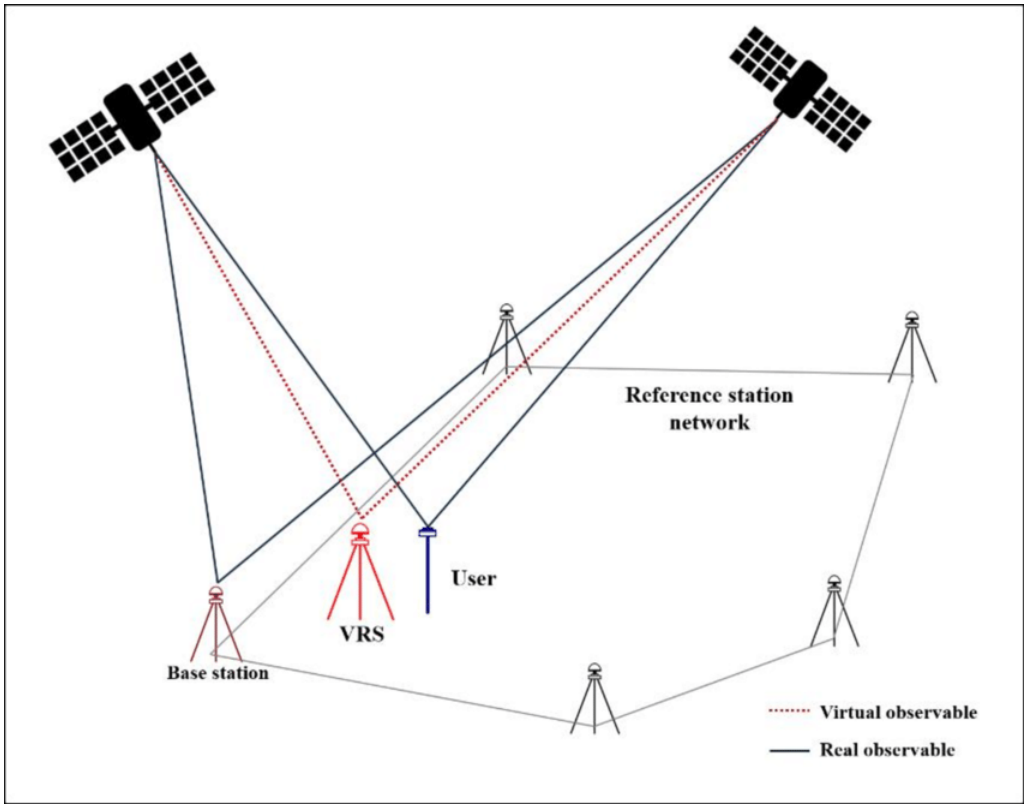


Figure 28: (Top) Difference between NTRIP Caster and (Bottom) VRS [37][38]

Figure 29 demonstrates the Base Rover setup.

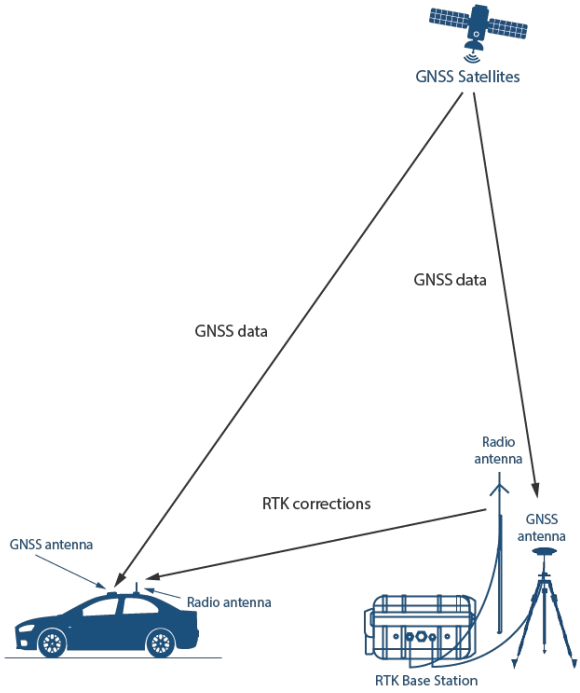


Figure 29: Demonstration of Base Rover setup

- **Bluetooth module**: Figure 30 shows the Bluetooth module (SKU AS-XBEE-BT-2.4-INT-00), which may be attached to the existing XBEE socket of the SimpleRTK2B module to provide a sleek and compact solution, costs approximately 29,00€. The default setting is 38'400bps, and it is compatible with all simpleRTK2B and simpleRTK3B boards, as well as cellphones, PCs, and other Bluetooth 2.0-capable devices. However, it is not compatible with iOS devices via Bluetooth, but it can be connected via WIFI, which has the advantage over Bluetooth that numerous devices can be connected.

Figure 30: XBEE Bluetooth module

- **OEM antenna**: The RTK multiband GNSS antenna shown in Figure 31 is connected to the SimpleRTK2B module and can work in base-rover/base-multi rover configuration,

supported positioning signal bands: (GPS: L1, L2), (GLONASS: G1, G2), (Bei Dou: B1, B2), (Galileo: E1, E5b), (QZSS: L1, L2), (SBAS: WAAS, EGNOS, MSAS and GA-GAN). Weight: 78g, Azimuth Coverage: 360 degrees, Maximum length: 76mm, Supply voltage: 2.5-5.5V, Price: 57,00€.

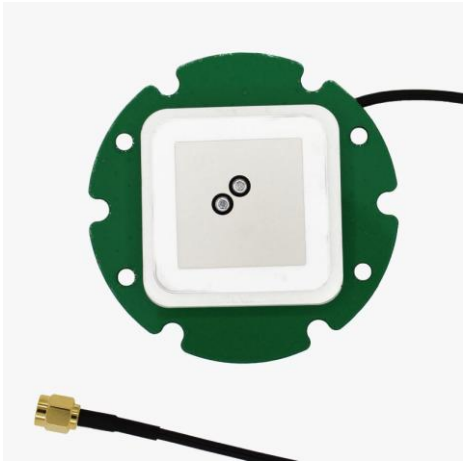


Figure 31: OEM antenna

- **Raspberry Pi 4 Model B**: The Raspberry Pi has unique specifications like 4gb of ram, 4 USB ports (2 which are 3.0 and 2 that are 2.0), 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, 2 × micro-HDMI ports, type C power supply and Gigabit Ethernet. In addition, the Raspberry Pi 4 Model B costs about 200€. As illustrated in Figure 32.

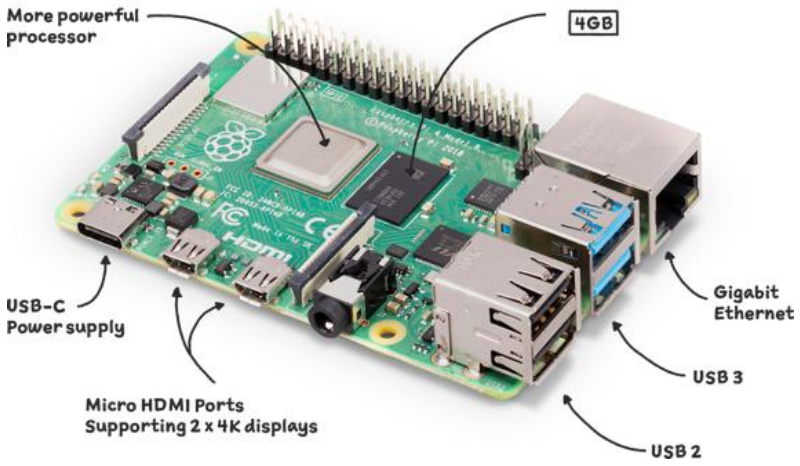


Figure 32: Raspberry Pi 4 Model B

- **Raspberry Pi 3.5inch Display**: The monito has 320×480 resolution, resistive touch control, it Supports any revision of Raspberry Pi (directly-pluggable) and the size perfectly fits the Raspberry Pi. Furthermore, this screen costs about 22€. Which can be seen in Figure 33.

Figure 33: 3.5inch Raspberry Pi screen (front and back)

Summary of the cost of different sensors used in the development of the MLRTK can be shown in table 4.

Table 4: The cost of sensors used within MLRTK.

Product	**Price €**
Velodyne VLP-16	2800
Raspberry Pi 4	200
SimpleRTK2B	230
Case Printing	100
6-Pin JST-GH to 6-Pin JST-SH	3
7404 Not Gate IC	2
DC-DC Boost Convertor XL6009 module	5
3.5inch RPi Display	22

Bluetooth module	29
OEM antenna	57
Total	**3,448 €**

3.2 Hardware Configuration

Configuration refers to the exact arrangement of a system or device's many components, settings, or features to improve its efficiency or operation. This may involve installing software, configuring settings, and connecting external devices or systems. The configuration process is essential for the correct operation of the different devices.

- **VLP-16 configuration**: The configuration of the VLP-16 is as follows:
  - At the start the VLP-16 was connected via the ethernet port to the computer and powered up.
  - Next, is to access the LiDAR using the Velodyne LiDAR web page under "192.168.1.201". Moreover, when connected correctly the PPS should synchronize and becomes "Locked".
  - At the end, when the GPS is connected and the message is transferred correctly, then the GPS position should be achieved as shown in figure 34.

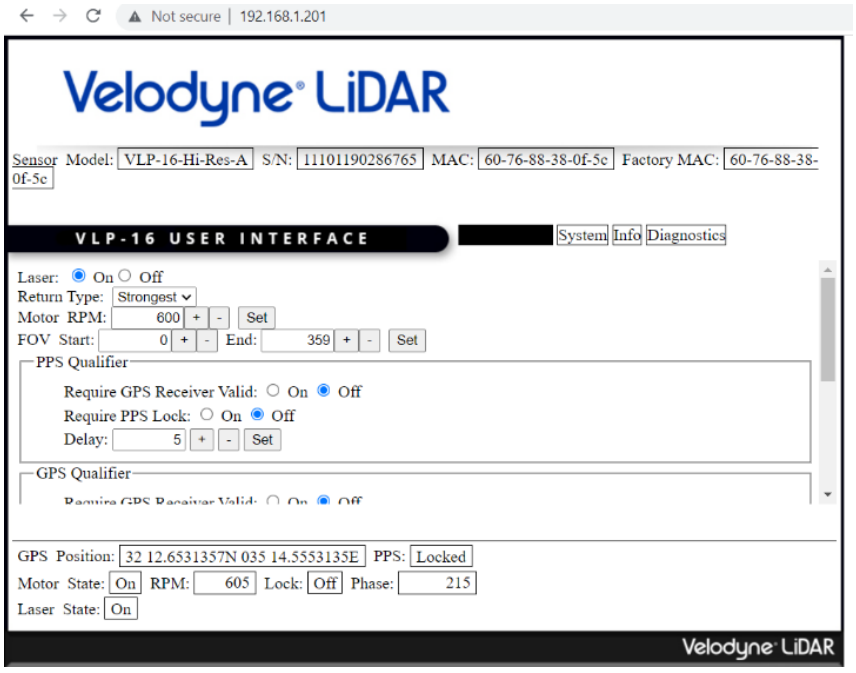


Figure 34: Velodyne configuration page

- **SimpleRTK2B configuration**: The SimpleRTK2B was configured as follows:
  - First step of setting up the configuration of the SimpleRTK2B is to download the Firmware and to upload it to the connected GPS using a USB cable via the U-Center software.
  - Then the simpleRTK2B configurations must then be loaded and modified to make it connectable to the VLP-16. Only the GPGGA and the GPRMC messages are accepted by the LiDAR, as shown in the VLP-16 manual.
  - Finally, the GGA message, however, was picked because it also has elevation values. Additionally, additional messages will be sent to the Bluetooth next to GPGGA via UART2 such as (GPRMC, GPGRS, etc.). One crucial step is to make sure that the NMEA Protocol is version 2.3, the baud-rate is 9600, and the main talker is changed to GP (GPS only) instead of GN (GNSS) as shown in Figure 35.

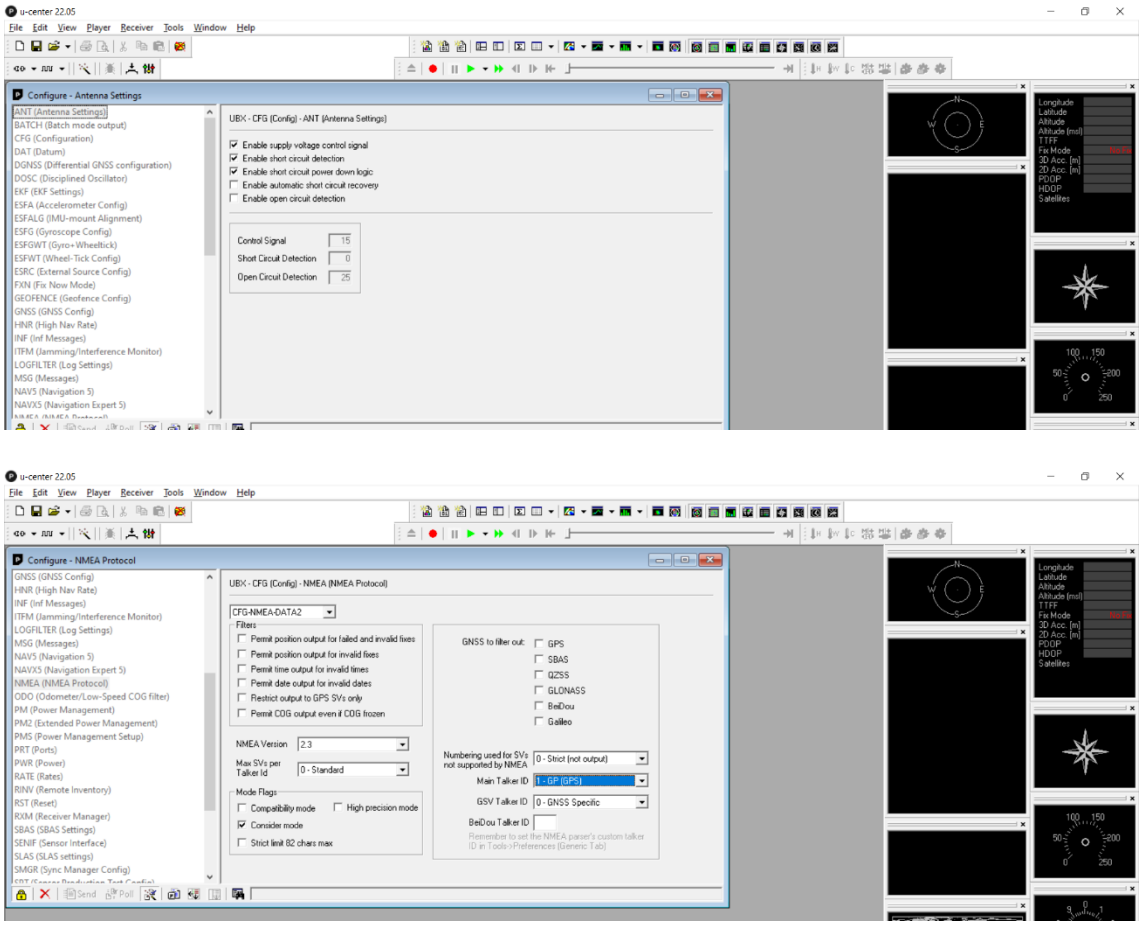


Figure 35: Two figures from the SimpleRTK2B configuration

- **Raspberry Pi configuration**: The Raspberry Pi was configured as follows and Figure 36 shows an example of the configuration:

  - First by connecting it to the monitor and the internet.

  - The next step is to launch the terminal using the command: "sudo apt install python3-dev python3-rpi. gpio git tcpdump" to install the necessary packages.

  - Furthermore, Cloning the rpi-puck-logger repository with "cd ~ git clone https://github.com/joshuabrockschmidt/rpi-puck-logger.git".

  - Moreover, "start.sh" was set up to run automatically on bootup, and data files will be timestamped and placed in the dump directory by using the command "sudo./setup.sh eth0 /logs/".

  - In the same way, new item was added to the Main Menu Editor from the Preferences menu that was named "LiDAR RTK." After that, the icon image was changed by going to Properties, clicking on the image, and selecting Change Image.

  - At the end using the codes "cd LCD-show/" and "./LCD35-show 180," the screen was turned by a 180 degrees at the conclusion of the Raspberry Pi's operation.

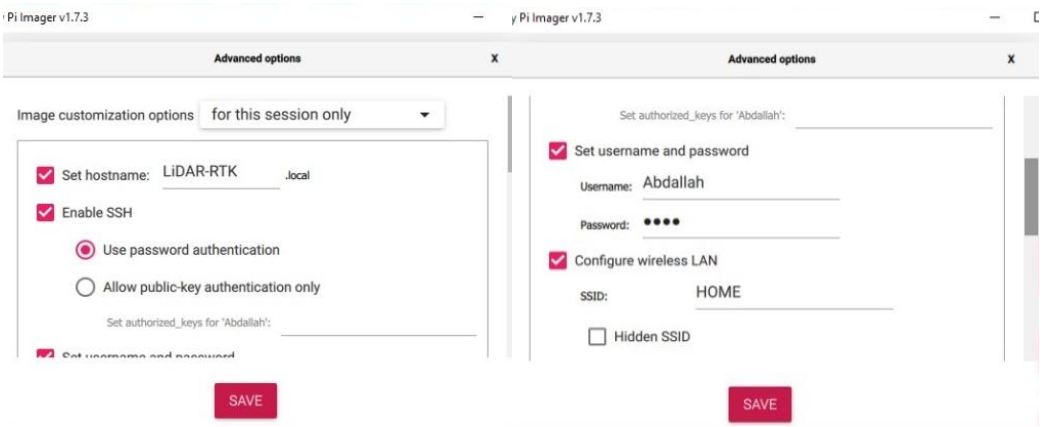


Figure 36: Two figures from the Raspberry Pi configuration

- **GNSS Base Receiver Base Configuration**: The process of configuring a GPS base station to supply accurate position and timing data to rovers. This procedure entails installing and configuring the hardware and software required to receive, process, and broadcast GPS signals, as well as any additional infrastructure and equipment required to support the base station and this process can be illustrated as follows:
    - Here the SimpleRTK2B with OEM multiband antenna were connected together with a power source. Then the base station was configured to be used to obtain correction for the LiDAR RTK project. Moreover, it was tested in an open sky area with clear view position that is previously known its coordinates. Also, several known locations were used to test and calibrate the functionality and the accuracy.
    - Then, to configure the SimpleRTK2B U-Center is required to adjust the GPS as base receiver. First step of setting up the configuration of the SimpleRTK2B is to download the Firmware and to upload it to the connected GPS using a USB cable via the U-Center software. Next, is to add the coordinates of the base station trough the TMODE3 (Time Mode 3) option in the configuration view window in the U-Center software. At the same time, Fixed position should be selected to add either XYZ or Lat, Long and Alt coordinate. The next step is to set up the NTRIP server/caster settings, to change the port and to set a username and password. Additionally, mount point settings should be established with Name, Identifier and Country code.
    - Next step is to transmit the GNSS data via the internet which can be done free of cost using the “Lefebure” NTRIP-Relay software. “Lefebure” NTRIP-Relay is a software tool that is used to facilitate the transmission of GNSS data over the Internet using the NTRIP protocol. “Lefebure” NTRIP-Relay is used to connect a GNSS receiver, such as a GPS receiver, to an NTRIP caster, which is a server that broadcasts GNSS data to clients over the Internet. The “Lefebure” NTRIP-Relay program functions as a bridge between the GNSS receiver and the NTRIP caster, enabling the transmission and reception of GNSS data over the Internet. To perform this the source caster settings should be modified. Beginning with adding the computer's IP address, port (which must match the port used in U-Center), username, and password.
    - To finish, it is also vital to include the base's Latitude and Longitude. Next, add the destination caster settings, beginning with the address, port, mountpoint

name, and password. Figure 37 shows the Base setup and casing created for this research.

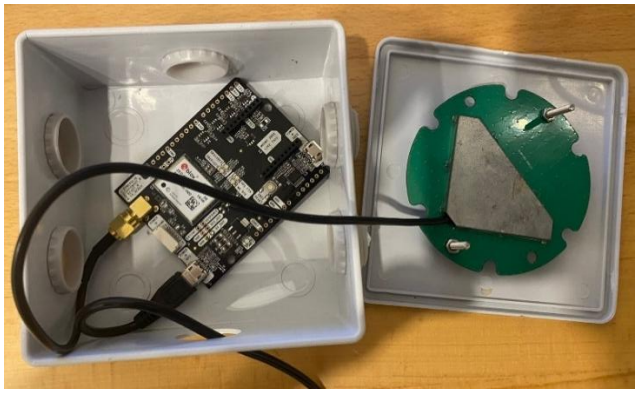


Figure 37: SimpleRTK2B Base Setup

3.3 Hardware Integration

Considerable effort has been taken to outline the integration of hardware components to build the Mobile RTK LiDAR device that can be used for data collection of both LiDAR observations and RTK positions. Figure 38 illustrates the wiring diagram of different components of the Mobile RTK LiDAR system.

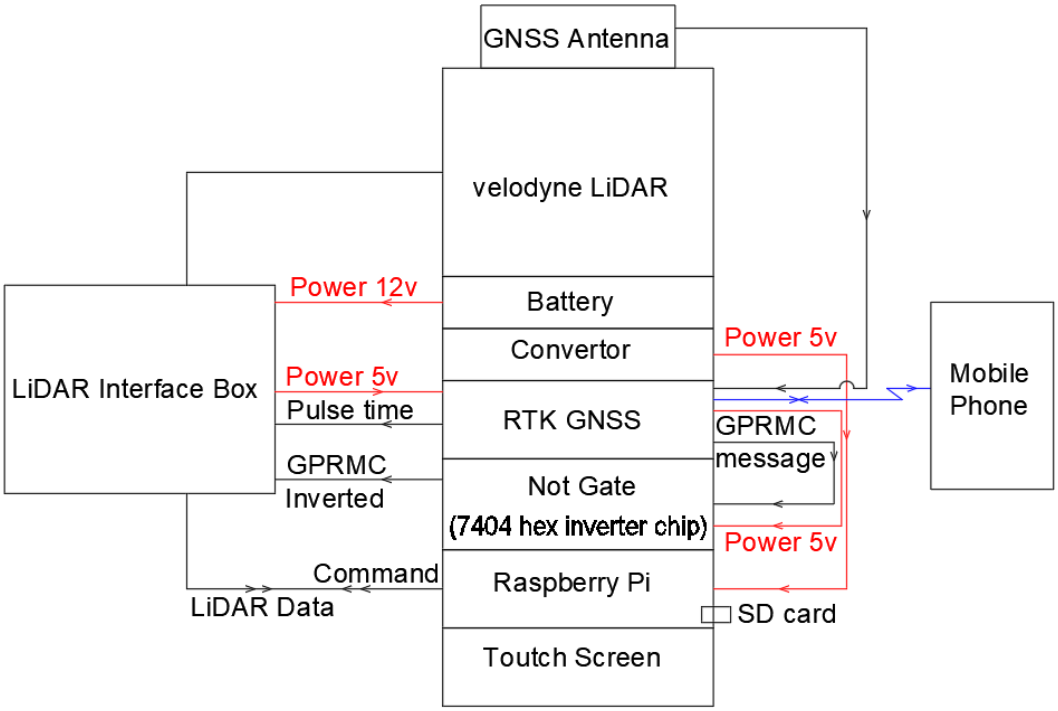


Figure 38: Mobile RTK LiDAR wiring diagram

The main components in this system are the VLP-16 LiDAR and the SimpleRTK2B GNSS receiver. In addition, these devices are integrated together with other devices to produce a powerful lightweight data collection tool. In general, the devices and the method used for the integration are illustrated as follows:

- **VLP-16 LiDAR and Interface Box**: This is the main component of the Mobile RTK LiDAR system. VLP-16 LiDAR connected with interface box that control the input power (proper voltage should be between 9 Volt and 18 Volt) to the device, the output data, and the input GPS data. The GPS pulsed data are used for synchronization of LiDAR data. Synchronizing to a GPS-supplied Pulse-Per-Second (PPS) signal provides the ability to compute the exact firing moment of each data point. In addition, LiDAR can utilize a once-per-second NMEA GPRMC or GPGGA sentence output from GPS device. The GPS data and synchronized LiDAR data are required by geo-referencing applications. The LiDAR device generates two types of packets: Data packets and Position packets. These packets are recorded in Packet Capture (PCAP) file in binary format. The data packet contains information such as distances, azimuth, elevation angle, intensity, and other data. Positional packets are sometimes referred to as telemetry packets, or GPS packets and contains NMEA GPRMC or GPGGA information.

- **RTK GNSS receiver**: The RTK GNSS receiver used within the developed device is the simpleRTK2B receiver (more information about this receiver can be found in Sensors Section 3.1). The native signal coming from the GPS receiver does not match the required polarity of the LiDAR sensor. Therefore, it is necessary to invert the GPS output signal using a 7404 hex inverter chip or equivalent circuitry. Within Mobile RTK LiDAR device, a special cable has been created to communicate between the LiDAR and the GPS device see Figure 39.

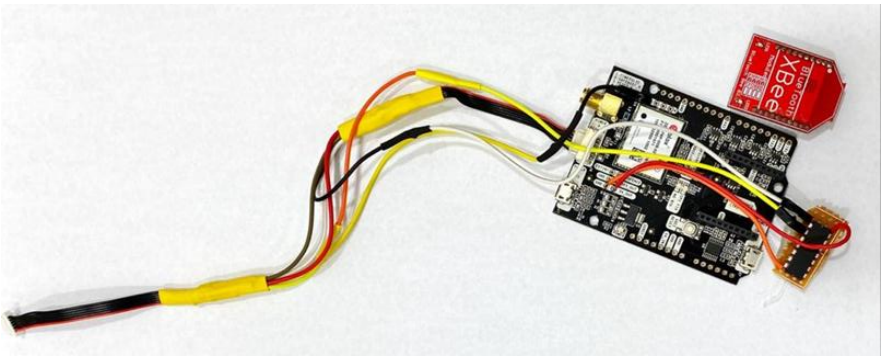


Figure 39: The GNSS receiver, 7404 Not Gate IC, Bluetooth, and cable.

The cable with 6-Pin JST-GH to 6-Pin JST-SH were cut and reconnected correctly to match the input and output of GPS and LiDAR devices as shown in Figure 40. There are 6 connectors on the GPS Pixhawk: 1:(5V_IN), 2:(RX), 3:(TX), 4,5:(Not connected) and 6:(GND). Furthermore, the GPS port in the LiDAR interface has 6 connectors, which are 1:(PPS), 2:(+5V), 3:(GND), 4:(RX), 5:(GND) and 6:(TX).

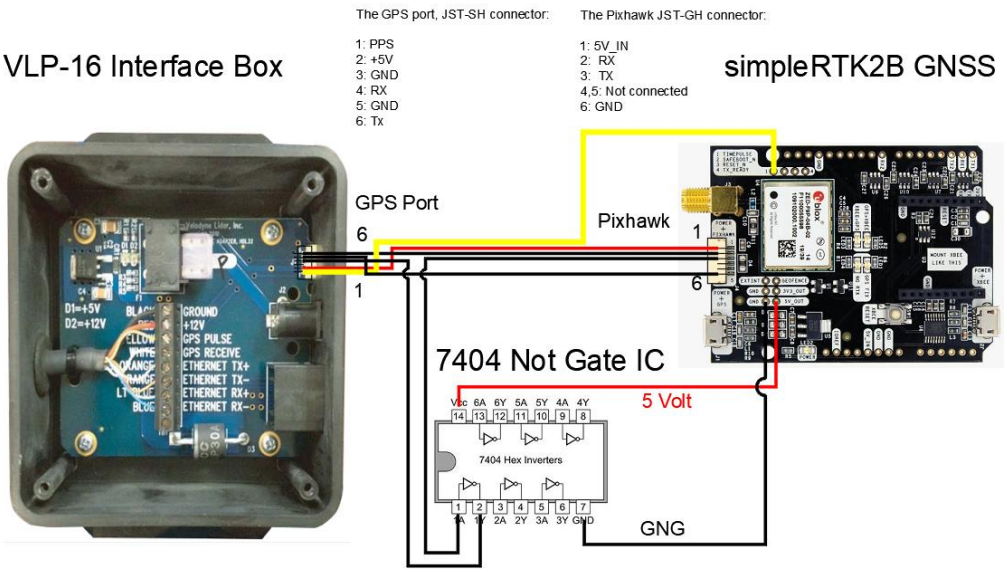


Figure 40: Wiring diagram of cable between simpleRTK2B GNSS receiver, 7404 Not Gate IC, and LiDAR interface.

With the simpleRTK2B board, the cable was attached to the Time Pulse pin (shown in yellow in Figure 40), 5 volts out pin, and Ground (GND) pin. Then the cable was rewired as follows: Pin 1 (5v In) in GPS connected with Pin 2 (+5v) of LiDAR, Pin 2 (RX) in GPS connected with Pin 6 (TX) of LiDAR, Pin 3 (TX) in GPS connected with Pin 1 (TX) of 7404 Not Gate IC, Pin 2 in 7404 Not Gate IC connected with Pin 4 (RX) of LiDAR, and Pin 6 (GND) in GPS connected with Pin 5 (GND) of LiDAR.

To ensure there was no appreciable offset between the devices, the GNSS antenna was mounted to the top center of the LiDAR. Only the offset between the antenna's and the LiDAR's centers will be considered.

- **The Raspberry Pi and touch screen**: The Raspberry Pi and its 64 GB micro-SD card are connected to a touch screen, and both devices are powered by a 5-volt supply. Moreover, the Raspberry pi was linked to the LiDAR interface box using an Ethernet cable.

An icon on the Raspberry Pi touch screen can be selected to begin logging LiDAR data as soon as the device is turned on.

- **Power supply**: The system was powered using 2200mA/11.1 V (3x3.7 V connected in serial) rechargeable lithium-ion batteries. The LiDAR interface box was connected directly to the batteries with an On/Off switch button that power the system. In addition, DC-DC Boost Convertor XL6009 module was used to output 5v for raspberry pi components (Figure 41).

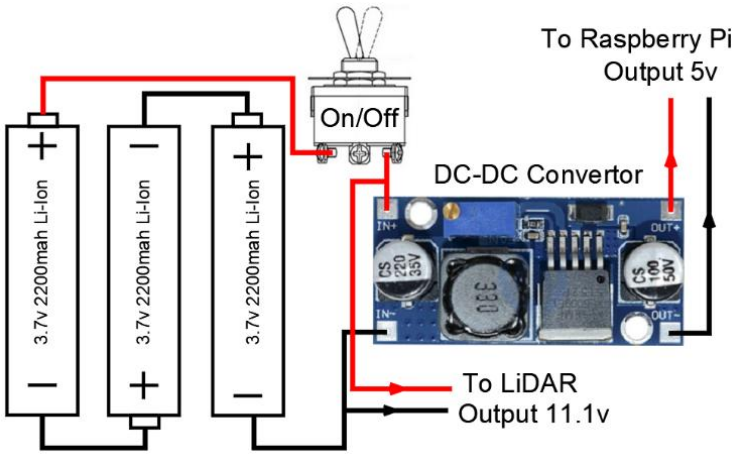


Figure 41: Wiring diagram of power supply

- **Housing Case and Separators**: The Raspberry Pi, GPS, DC-DC convertor, and batteries were placed within a box that was designed and plotted using a 3D printer to be used as a housing case for the Mobile RTK LiDAR system Figure 42.

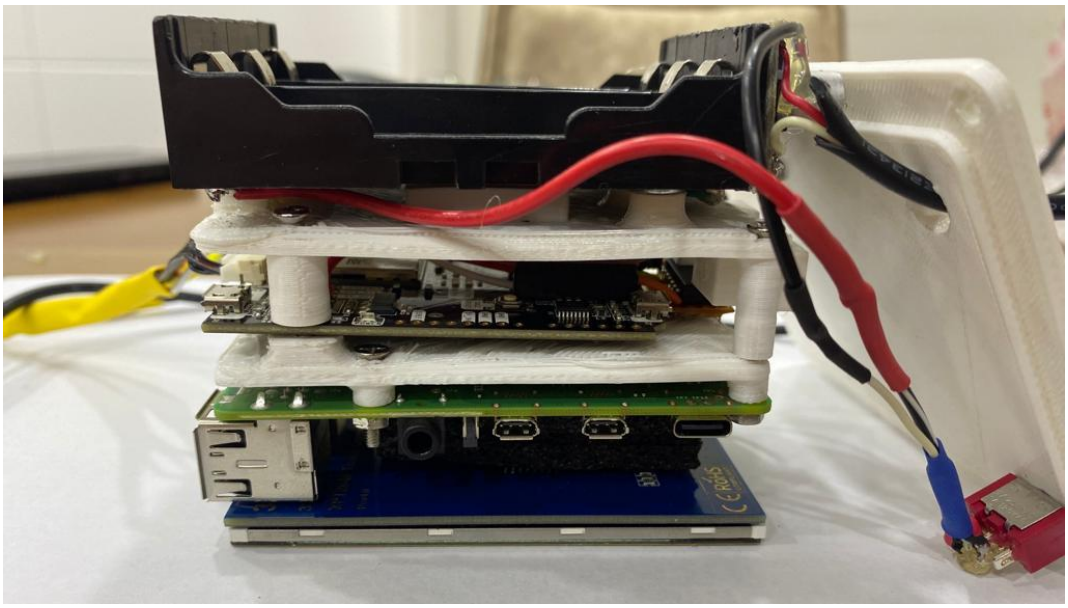


Figure 42: Internal components and separators of the MLRTK device

The housing case has dimensions of (10.8*9.7*8.5 cm). The package also comes with two separators: one divides the GPS from the battery holder, and the other separates the raspberry Pi from the GPS. The proposed product consists of 3 holes in the box for the antenna, 4 USB ports and the ethernet port. Also, it weights 680g including 3*18650 li batteries that weights (45g) each. Moreover, the final products weights 680g, 130g for the holder and 1015g for the LiDAR, interface box including 3m wire and 1m ethernet wire (so about 1.825kg in total). The LiDAR RTK can last up to 120 minutes of scanning time operating the LiDAR, GPS and the raspberry pi. However, it is worth mentioning that the ZEB Horizon weights 1.45kg alone and 1.4kg datalogger weight including the batteries (2.85kg in total) [38]. Also, the Paracosm px-80 weights about 2.9kg excluding external batteries and can scan up to 40 minutes [39]. Figure 43 shows the final device developed for this research.

Figure 43: LiDAR RTK Final Outcome

Overall, the designed MLRTK device enables measurement collection at a higher degree of complexity while maintaining simplicity and being fully synchronized with RTK data. This

lightweight, user-friendly device is now prepared for data collecting. The prior design, however, shows that it is feasible to create a low-cost LiDAR system, and if it were to be made into a commercial product, it might be enhanced.

The developed MLRTK device can be utilized in the field for data collection as follows:

- MLRTK device can be mounted on a surveying pole.
- Switch the power on.
- Open the NTRIP client app on your smartphone.
- Pair GNSS receiver via Bluetooth.
- Wait for RTK fixed solution.
- Launch the Raspberry Pi's LiDAR logging.
- Start data collection.

For data collection, the Stop & Go strategy is preferred. Moreover, begin data collecting at a specific location, and end data collection at the same location (close the loop). While the RTK data can be sent through Bluetooth from the mobile phone, the gathered LiDAR data (PCAP file) can be downloaded from the Raspberry Pi. Now, the data are ready for post-processing which be discussed in the next chapter.

4 Development of LiDAR RTK Software

This chapter will discuss the relevant software utilized in this study and why it was chosen. Additionally, it will discuss how to use and utilize the created LiDAR RTK software.

4.1 Related Software's

The usage of software has become pervasive in nearly all fields, and the selection of the appropriate software can have a significant impact on the success of a project or activity. There are numerous software solutions accessible for study or work, each with its own set of features and capabilities. Therefore, these programs were chosen after extensive research and comparisons with other programs.

4.1.1 Velodyne "Veloview" software

"VeloView" is a free open-source "ParaView"-based software developed by "Kitware" [40]. It is designed for the 3D point cloud analysis and visualization of raw Velodyne LiDAR sensor data such as VLP-16 and other sensors. There were several editions of this software, and version 3.5 was selected since it was believed that it can perform SLAM-based mapping, with and without GPS/IMU. However, it was found that this function is not yet fully available. SLAM processing is not supported by the more recent "VeloView" versions, such as 5.1. In addition, "LidarView" has lately replaced "VeloView", which had SLAM processing capabilities. Similar to "VeloView", there are several editions of "LidarView" and the latest version is v4.3.0 released on (18/01/2023) [41]. This version of "LidarView" was tested and the results of this test will be discussed later in this section.

In this study, "VeloView" software was utilized for three tasks: first, to clean the unwanted data, second, to build SLAM trajectories, and third, to compare the data that LiDAR RTK software had extracted from the binary PCAP file. First, the PCAP file should be cleaned from the unwanted data before performing SLAM. These unwanted data are the data before the RTK fixed position obtained and the last data of the PCAP file during switching the Mobile RTK LiDAR device off. These data should be deleted from the PCAP file, and the cleaned file can be used within LiDAR RTK software.

Second, to produce SLAM trajectory, the new software "LidarView" was firstly tested to process the PCAP file. The software was run to produce a trajectory path using SLAM default values and

using all frames in the PCAP file. The result in Figure 44, shows that the software was not able to produce a reasonable trajectory and it fails after testing point 5.

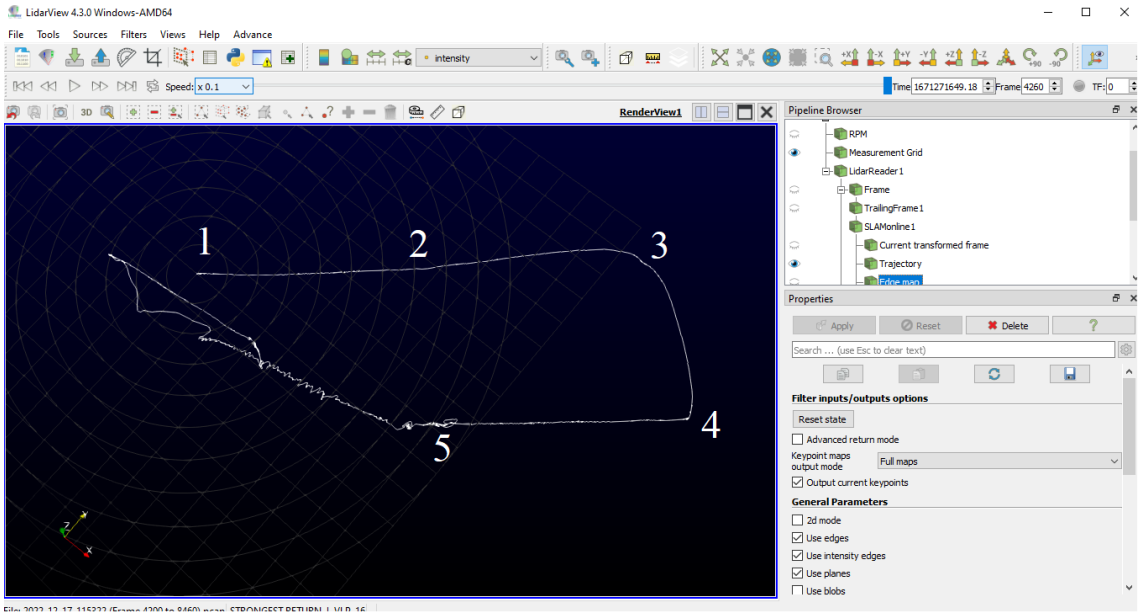


Figure 44: Slam Trajectory obtained from “LidarView” with defaults SLAM values.

To overcome this error, the PCAP file was processed again after modification on the SLAM options that can be enabled by the advanced properties of the SLAM filter which are enabled by toggling the little gear wheel in the Properties panel. The SLAM parameters can be modified for different environments as follows [42]:

- Outdoor scene
  - Keyframe distance/angle threshold: 0.5-1 m distance, 2-5° angle.
  - Edges/Planes map resolution: 30 cm for edges, 60 cm for planes.
- Indoor scene
  - Keyframe distance/angle threshold: 0.1-0.5 m distance, 5° angle.
  - Minimum distance to sensor: 0.5 m
  - Edges/Planes map resolution: 20 cm for edges, 30 cm for planes
  - Rolling grid resolution: 3 m.
- Poor geometric scene or scene with some strong invariance: corridor, fields, highway, forest ect.
  - Keyframe distance/angle threshold: 0 m distance, 0° angle (disabled).
  - Use Blobs: enabled.
  - ICP-Optimization iterations: 4

- Edges/Planes map resolution: 20 cm for edges, 30 cm for planes

It was noticed that the default results match the outdoor scene, therefore, the SLAM properties modified to poor geometric scene. Similar results to previous processing have been achieved, and the software fails after test point 5 (Figure 45).

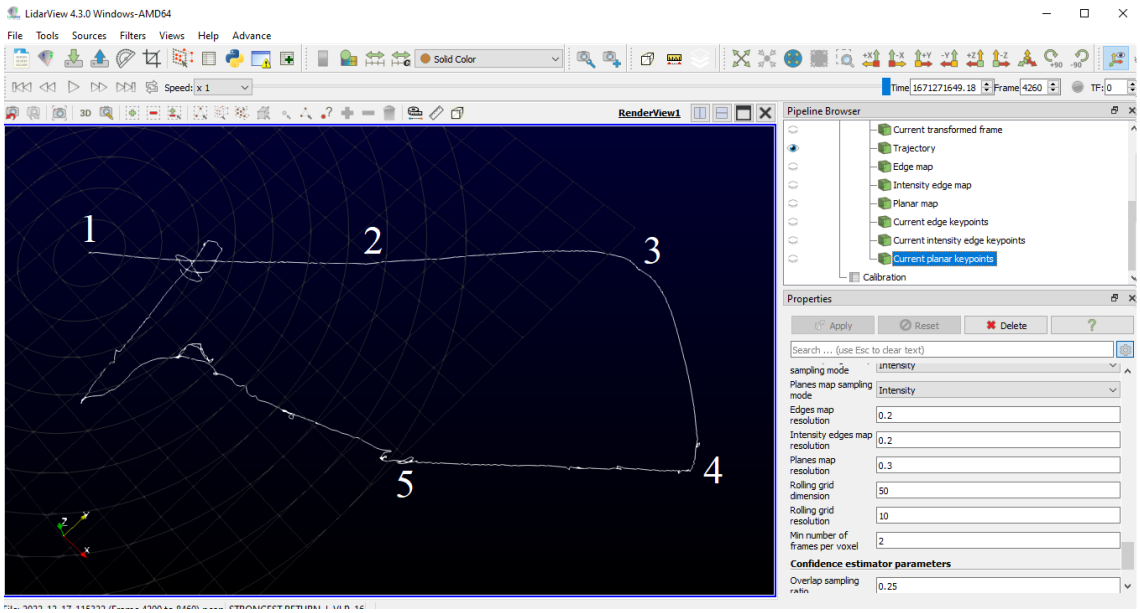


Figure 45: Slam Trajectory obtained from “LidarView” with modified SLAM values.

To overcome the errors in the SLAM trajectory, “VeloView” 3.5 was then tested to produce trajectory file and maps. The advanced settings in “VeloView” should be activated to perform SLAM (Figure 46).

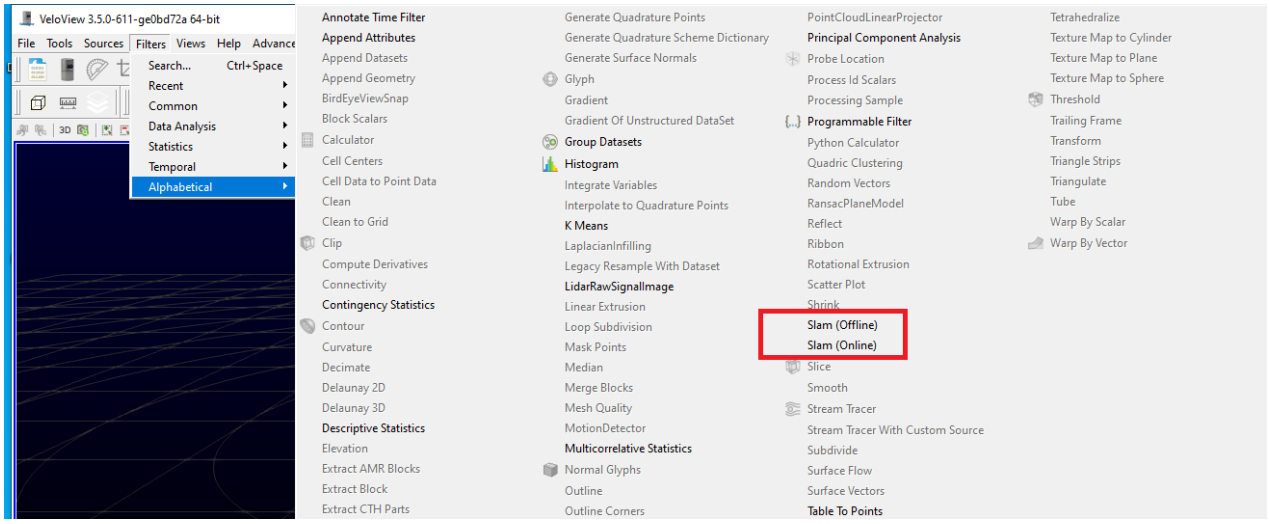


Figure 46: “VeloView” software with advanced settings and SLAM options enabled.

The software can process all available frames or selected time interval such as 1 second (x 1). Unlike, “LidarView”, there is no advance processing options for the SLAM processing within “VeloView”. The PCAP file of Palestine test data were post-processed using “VeloView” software and it was able to perform SLAM successfully. The SLAM trajectory resultant from the post-processing shown in Chapter 5.2.1.

The third use for “VeloView” is as a testing tool for LiDAR RTK software in development. This will be evaluated after the LiDAR RTK software has been finished.

4.1.2 U-Blox U-Center

U-Center is an extremely user-friendly GNSS evaluation program used to configure U-Blox GNSS receivers. In this project, the U-Center software was used to configure both the base and the rover (Figure 47). In addition, this software identifies the mount point that will be casted by the “Lefebure” NTRIP Relay Software.

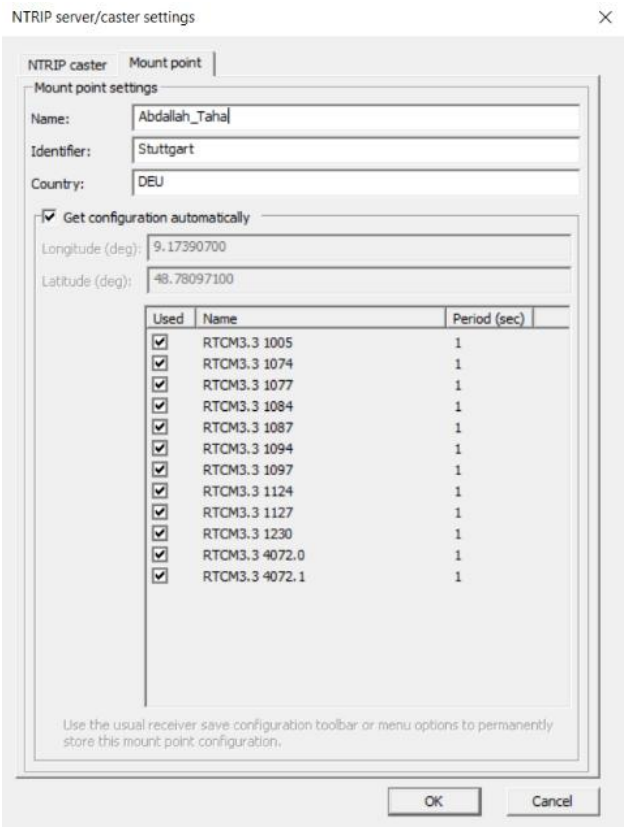


Figure 47: U-Center Mount Point Identification

4.1.3 NTRIP “Lefebure” Relay

This application combines NTRIP Client and Server functionality. It is designed to connect to a caster to retrieve data and then send that information to another caster. This program was used in this research to cast the GPS connections and send them to the receiver via an internet connection See Figure 48.

Figure 48: “Lefebure” NTRIP Relay Connection

4.1.4 NTRIP Client for Android

This application is used to connect to a high-accuracy GPS or GNSS receiver, transmit DGPS or RTK correction data to the device through NTRIP over the phone's Internet connection, and receive correction data from the receiver. Record NMEA location data from this receiver to an SD card file. Provide access to NMEA position data from an external receiver to other Android applications see Figure 49.

Figure 49: NTRIP Client for Android in field

4.2 LiDAR RTK Software

LiDAR RTK is a software developed in this research using Visual studio (Vb.net) language for LiDAR data and GPS RTK data. With features like Add RTK Data, Add Trajectory Data, Process, Export, open and the Media buttons makes it easy to manage projects and delegate tasks. Currently this software is used for post-processing however, it can be converted to real time processing due to usage of segmentation.

4.2.1 Purpose of the software

The software can read the outcome of LiDAR (PCAP) packet which is in a binary format. Also, it can read the trajectory file which includes the (X, Y, and Z) coordinates and both the time stamp and the ID. Moreover, the software can export PCD files, Object files and Trajectory. Additionally, the software has several options dealing with the frames starting with the play button which allows the user to see all the frames with its number. Next is the stop button which stops playing the frames. Also, there are two buttons (previous and next) to navigate through the frames one by one. Besides, the ability to visualize between different classes of data (LiDAR data, GPS data, Trajectory data, RTK trajectory, Frames info and SLAM trajectory).

4.3 Software Libraries Incorporated into the LiDAR RTK Software

This section provides a brief description of the various software libraries incorporated into the LiDAR RTK Program.

4.3.1 OpenTKLib

The “OpenTKLib” is an open-source user control that was developed by Edgar Maass on 2016 [43]. It is a .NET utility that uses “OpenTK” which encapsulates and extends OpenGL. It is a powerful library for displaying 3D points very fast and can be used for handling point cloud. Several features can be performed on the user control such as zoom in and out, pan, rotate, and change color of the model or background. However, point clouds needed to be converted into OBJ format to be loaded into the “OpenTKLib” user control. OBJ (or .OBJ) is a 3D model format developed by Wavefront Technologies and used by other 3D graphics applications [44]. The OBJ file format is a simple data-format that represents 3D geometry alone, namely, the position of each vertex, the UV position of each texture coordinate vertex, vertex normal, and the faces that make each polygon defined as a list of vertices, and texture vertices. A simple way to create OBJ

file is by creating a text file with format in each line as follows: letter “v” which means vertex, space, x coordinate, space, y coordinate, space, and z coordinate.

Other features that available within this library but not currently used are such as loading multiple models, moving one model in comparison to the other in space, and most importantly, the point cloud registration.

4.3.2 HighPerformanceListView

The “HighPerformanceListView” is code written to demonstrate a technique for displaying very large quantities of data in a visual control, with very little performance degradation. It is a composite control that compresses a .NET “ListView” control and a “VScrollBar” control [45]. This control makes use of an object that implements the “HighPerformanceListViewProvider” interface to programmatically provide the data in a general manner. Additionally, this object supplies the additional functionality needed by the List-View control to determine the column headers being displayed, and to provide “ListViewItems” that represent one item of data.

The strategy for displaying large quantities of data is to have a list control that will only contain data items that are visible to the user, not the entire data set. The existing .NET “ListView” component is used, coupled with a “VScrollBar”, which is manually adjusted to reflect the quantity of data being represented. As “ListViewItems” move out of view, they are deleted, and as new ones come into view, they are added, so that the list only contains a maximum of the number of items that can be viewed by the user.

4.3.3 Ribbon

The ribbon that is used in this research is an open-source project created by Jose Menendez Poo [46]. The ribbon class library can be added via “Nuget”, and panels can be added as needed. Several items can be added to the panel such as Button, Text-Box, Combo-Box, Label and other items as shown in Figure 50.

Fehler! Verwenden Sie die Registerkarte 'Start', um Heading 1 dem Text zuzuweisen, der hier angezeigt werden soll.

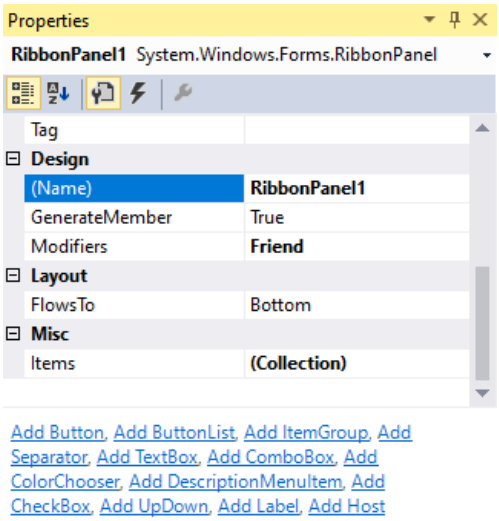


Figure 50: Items that can be added to the panel in the Ribbon

4.3.4 Coordinate-Sharp library

Coordinate-Sharp is a simple .NET library that is created to assist with geographic coordinate conversions, formatting, and other calculations [47]. It is used within the LiDAR RTK software to convert geodetic (Latitude and Longitude) coordinates to UTM (East and North) coordinates.

4.3.5 Math.NET Numerics library

Math.NET “Numerics” is an opensource numerical library for .NET, aiming to provide methods and algorithms for numerical computations in science, and engineering [48]. It is mainly used within LiDAR RTK software for performing Polynomial interpolation.

4.4 LiDAR RTK Functions

The LiDAR RTK software was developed in this research using Visual studio (Vb.net) language for the purpose of producing smoothed transformed trajectory file that can be used for georeferencing point cloud data and hence producing georeferenced 3d maps. The software is mainly used for post-processing LiDAR data, RTK NMEA data, and performing an integration between the SLAM trajectory data and RTK data see Figure 51.

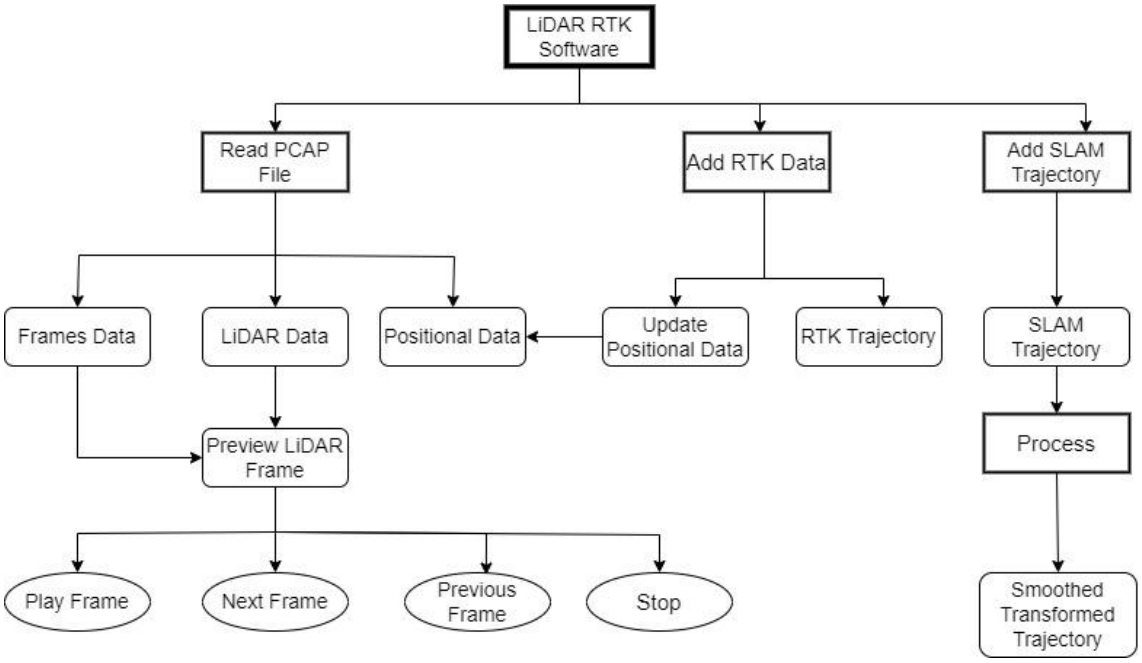


Figure 51: LiDAR RTK software flowchart

There are other features that were added to the software such as LiDAR data visualization, converting LiDAR coordinates to .obj format, exporting output trajectory files and others. Moreover, with click few buttons like Read PCAP file, Add RTK Data, Add Trajectory Data, Process, and Export, makes it easy to manage projects and delegate tasks. Currently this software is used for post-processing however, it can be developed to a real time processing software. All functions used in the development of this software are attached in Annex III. This section describes the functions used within the LiDAR RTK software in detail.

4.4.1 Global Variables

A global variable is a programming language construct, a sort of variable declared outside of any function and available to all functions throughout the program. Since all functions have access to them during program execution, they are often stated on top of all other functions and kept to a minimum. Several variables on the LiDAR RTK program have been declared as global variables. The complete variables are shown in Annex III; however, the following are some of the key variables for illustration:

- VLP-16 Lidar vertical angles: Public SENSOR_VERTICAL_ANGLE_VLP16 As Integer () = {-15, 1, -13, 3, -11, 5, -9, 7, -7, 9, -5, 11, -3, 13, -1, 15, -15, 1, -13, 3, -11, 5, -9, 7, -7, 9, -5, 11, -3, 13, -1, 15}

- Interpolated LiDAR Azimuth / horizontal angle: Public “azimuth_interpolated” As Double = 0
- Value to convert degree angle to rad angle: Private “deg_to_rad_coeef” As Double = (“Math.PI” / 180.0)
- Number of bytes in frame: Const BYTES_IN_FRAME As Integer = 1206

4.4.2 Reading PCAP File

As mentioned in section 2.7, the Velodyne VLP-16 LiDAR provides only the raw UDP data packets via its Ethernet port, and the device does not generate the point clouds. Processing steps within this function are illustrated in Figure 52.

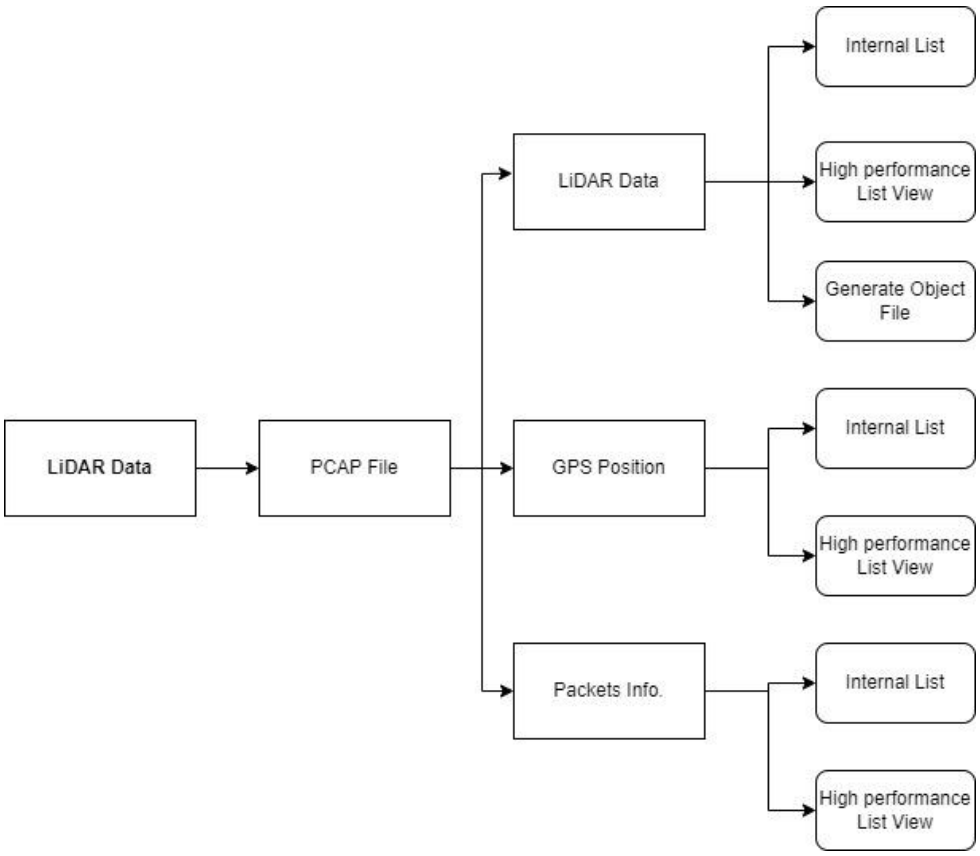


Figure 52: Processing steps for extracting data from PCAP file.

As shown on Figure 52, the read PCAP function includes three main functions as follows:

1. **Packets Info**: The software starts with processing the data packets to find out frames’ information using “Extract_Packets” function. These data are as follows: Frame ID (start from 0), Packet From, Packet To, Time Stamp From, and Time Stamp To. This

information is then used for extraction the information of specific frame, and converting it to .obj file. Within “Extract_Packets” function, if the current Azimuth is less than the previous Azimuth, then it is a new frame. This means that the sensor completes a rotation, crossing the zero-azimuth angle. In general, it was found that the frame contains around 90 packets. The packet info data are stored in an internal list called “FramesDataPoints” which is a List of Packets Data Item (data as explained previously). In addition, the data are also stored in a high-performance list view (“ListViewPackets”) for visualization purposes.

2. **LiDAR Data**: The function “Extract_LiDAR_Data” (Current_Frame) is used to extract the current frame such as frame 0 when start extracting data. Data extracted are such as Time Stamp (with Time Stamp offset), Azimuth angle (using interpolation), Distance, Reflectivity of the object (Intensity), and Vertical angle. In addition, the cartesian coordinates x, y, and z of each point in frame coordinate system (0,0,0) is calculated as well. Frame coordinate system is that the origin point of each frame is always (X=0, Y=0, Z=0). Direction of LiDAR (Orientation). The sub functions that used incorporate with “Extract_LiDAR_Data” are: “Get_Azimuth”, “Azimuth_Interpolation”, “Get_Distance”, Get_TimeStamp, calc_X_Coordinates, “calc_Y_Coordinates”, and “calc_Z_Coordinates”. The extracted data and the calculated coordinates are stored in an internal list called “LiDARDataPoints” which is a List of LiDAR Data Item (data as explained previously). In addition, the data are also stored in a high-performance list view (“ListViewLiDAR”) for visualization purposes. The calculated coordinates are stored in a .obj file and loaded into “OpenTKLib” control to be displayed as points cloud using “LoadModelFromFile(input_File-Name)” function. For evaluation the accuracy of extracting the LiDAR data, same file was opened with “VeloView” software and the results of extracting the data are shown in Figure 53.

Fehler! Verwenden Sie die Registerkarte 'Start', um Heading 1 dem Text zuzuweisen, der hier angezeigt werden soll.

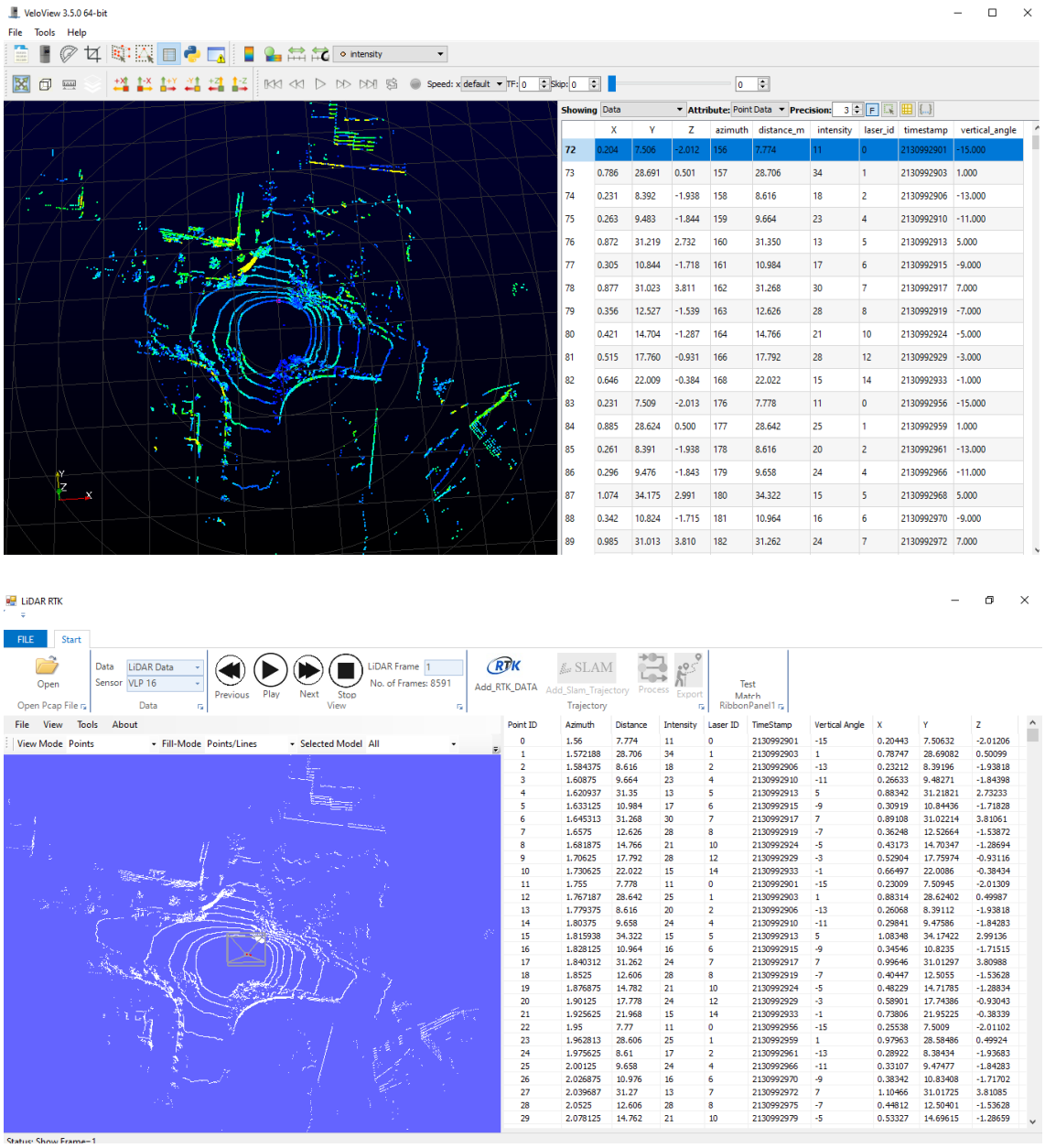


Figure 53: (Top) Results of extracting LiDAR data using VeloView software (Bottom) and LiDAR RTK software.

As indicated in Figure 53, LiDAR data at TimeStamp 2130992901 in VeloView (Distance=7.774m, Azimuth=156˚, Intensity=11, and Vertical angle =-15) and calculated coordinates (X=0.204, Y=7.506, and Z=-2.012) are exactly same values in the LiDAR RTK software. In addition, when comparing the of shape points cloud in both software, they are very similar. That expresses the ability of the developed software to process LiDAR data accurately.

For displaying the points cloud, additional capabilities, such as media buttons, have been added to the LiDAR RTK software. For example, play button shows the frames one by one continuous mode. The function RibbonButton_Play_Click calls the function Extract_LiDAR_Data(Frame_Name) which extracts LiDAR data of specific frame, save a .obj file of this frame, and increase the frame counter. Next, Previous, and Stop functions are also added to the software which each perform specific task such as Next shows Next frame, Previous shows previous frame and Stop terminate displaying the continuous playing.

3. **Positional Data**: The function “Extract_GPS_Data” is used to extract GPS data from NMEA GPRMC or GPGGA message, that is stored in PCAP file as positional data. This function uses a sub functions “ParseGPRMC” and “ParseGPGGA” . The extracted GPS data are as follows: GPS Time, Latitude, Longitude, Altitude (available on GPGGA message only), Quality, Time Stamp, Heading (available on GPGRMC message only), and Speed. The extracted Time Stamp should not used at this point, but the minutes and seconds part of the time supplied by the GPRMC sentence should be used instead. This time should be then converted into a top-of-the-hour (TOH) counter value, which is the number of microseconds that has elapsed since the top of the hour. Hence, Time Stamp is computed from the minutes and seconds of the GPS time as follows: for GPS time (18:42:30), minutes = 42+30/60 which equal to 42.5 minute. The Time Stamp = 42.5 * 60,000,000 which equal 2,550,000,000 microseconds, which should be used as the GNSS Time Stamp. Furthermore, the extracted data are stored in an internal list called “GPSDataPoints” which is a List of GPS Data Item (data as explained previously). In addition, the data are also stored in a high-performance list view (ListViewGPS) for visualization purposes. For evaluation the accuracy of extracting the GPS data, same file was opened with Wireshark (Version 4.0.1) software and the results of extracting the data are shown in Figure 54.

Fehler! Verwenden Sie die Registerkarte 'Start', um Heading 1 dem Text zuzuweisen, der hier angezeigt werden soll.

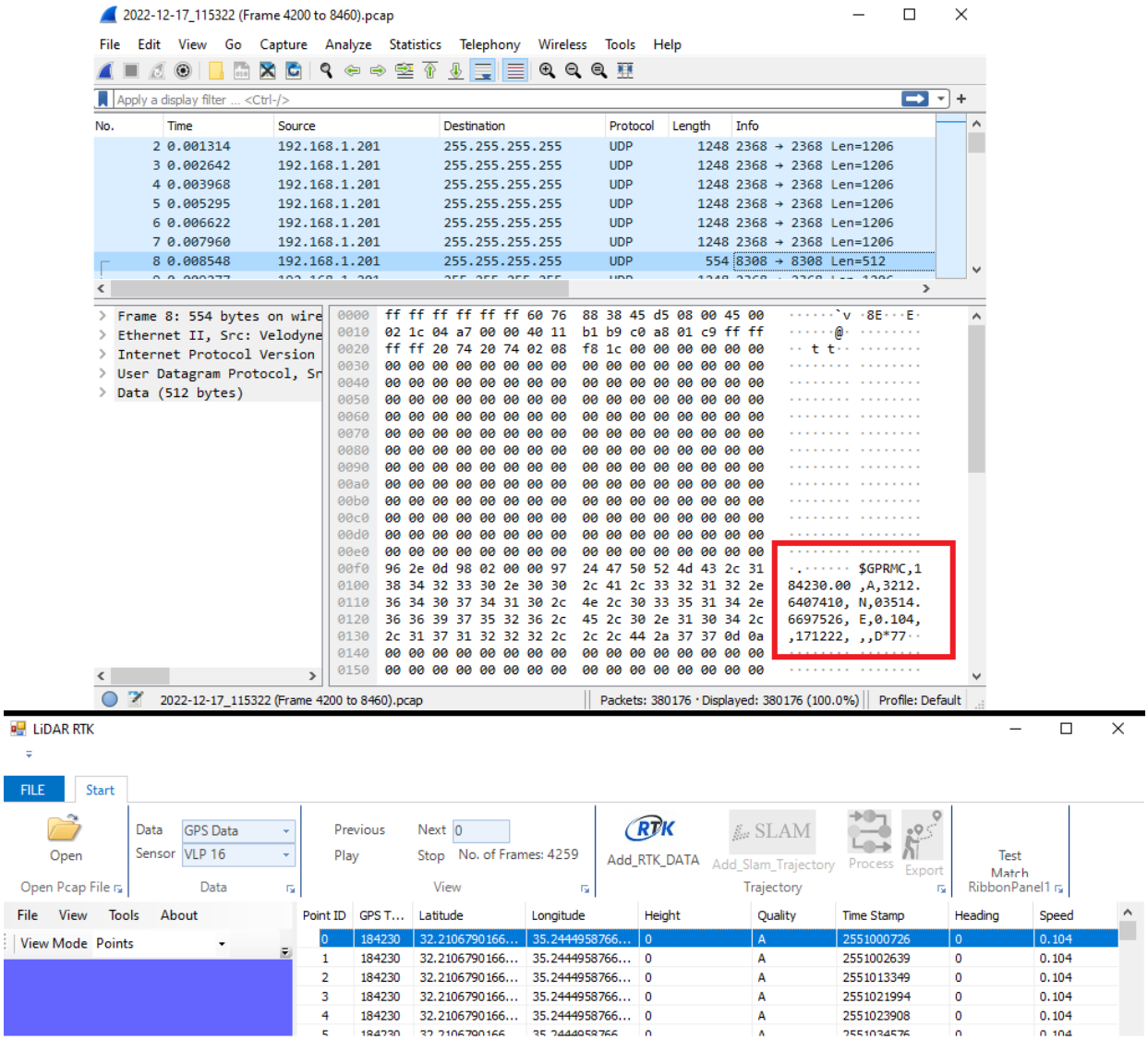


Figure 54: Results of extracting GPS data using Wireshark software (top) and LiDAR RTK software (down).

As can be seen from Figure 54, the GPS Time (184230), the position quality (A), and speed (0.104) are the same in both software. However, there are differences between the Latitude and Longitude, and this is due to converting these values to decimal degrees in LiDAR RTK software. When converting these values to decimal degrees, they are matching. That expresses the ability of the developed software to process GPS data correctly.

4.4.3 Add RTK Data

The RTK data are stored in the log file on NTRIP Client software during the data collection using the Mobile RTK LIDAR device. The data is a comma separated list of ascii data. Processing steps within this function are illustrated in Figure 55.

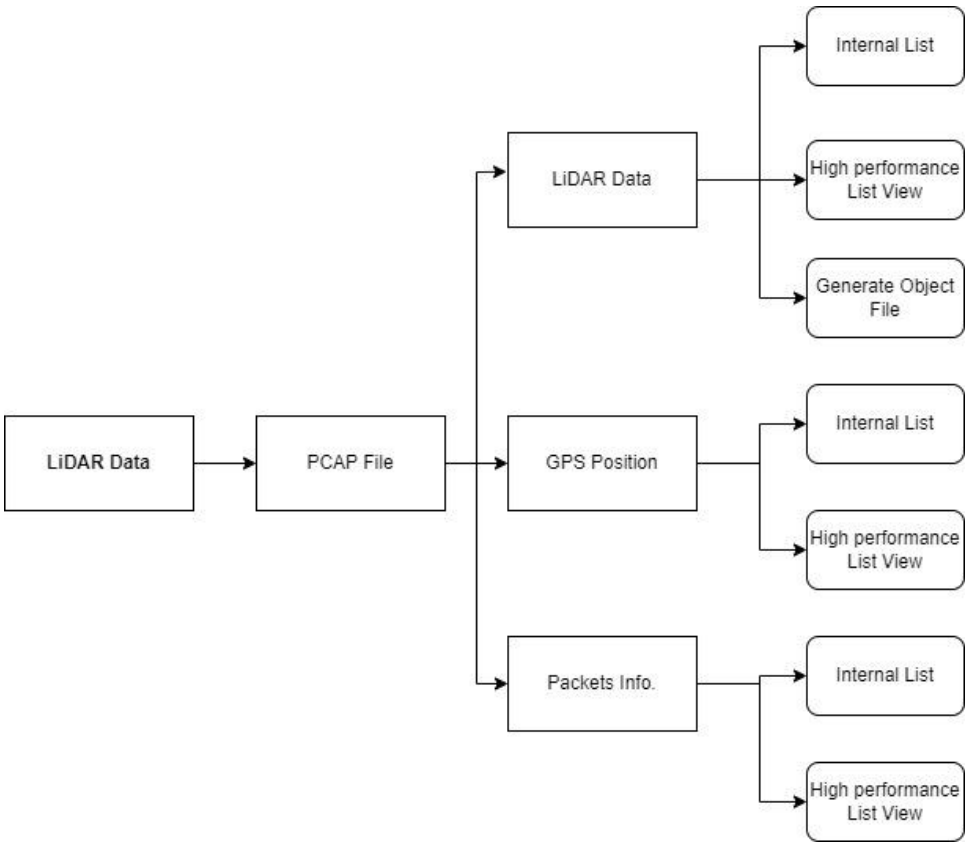


Figure 55: Processing steps for extracting RTK data from NMEA file.

As shown on Figure 55, the Add RTK data function mainly perform two tasks: update GPS data and RTK Trajectory as follows:

1. **Update GPS Data**: This option reads the NMEA file and updates GPS information using GPGGA message. Using Read_NMEA (File Name) function, all GPGGA messages are first filtered and stored in a temporary list called GGAListbox (List of String). Then, GPS data (GPS Time, Latitude, Longitude, Altitude, and Quality) are extracted from the GPGGA message using ParseGPGGA function. After that, the Latitude and Longitude are converted to UTM coordinates using CoordinateSharp library and the results stored with other extracted GPS data in a temporary list called TempNMEAList. The Positional data extracted from the PCAP file (GPSDataPoints) as explained in Section 4.4.2 are

updated by adding the RTK solution type (for example RTK fixed solution), adding Elevation, UTM East and UTM North information. The high-performance list view (ListViewGPS) is also updated using same information added as mentioned previously.

2. **RTK Trajectory**: The most crucial information is the RTK Trajectory Data since it can be utilized to modify the SLAM Trajectory. The function AddGPS2RTK_Traj is used to produce the RTK Trajectory from the updated list GPSDataPoints. However, all duplicate data as well as all data without RTK Fixed solution will be eliminated. The horizontal distance and Azimuth are calculated between sequence points and these data with other data are stored in an internal list called RTK_DataPoints. So, the RTK Trajectory data contains the RTK fixed solution data which are as follows: GPS time, UTM East, UTM North, Height, Time Stamp, Speed, Distance, Heading and Azimuth. Like other data, RTK Trajectory data are stored in a high-performance list view (ListViewRTK) for visualization purposes.

4.4.4 Add SLAM Trajectory

In this research, after processing the LiDAR data, SLAM Trajectory was produced using the VeloView program. The SLAM Trajectory file is an ascii file that contains information as follows: Time (Time Stamp), Rotation angles (Rx (Roll), Ry (Pitch), Rz (Yaw)), and coordinates (X, Y, and Z) for each frame. Unlike the RTK trajectory, SLAM Trajectory uses an arbitrary coordinate started from (X=0, Y=0, Z=0) of the first frame and these coordinates increased according to the Trajectory path. These coordinates are in the same coordinate system of the point cloud of Planner and Edge maps obtained from VeloView during the SLAM processing. Therefore, when the SLAM trajectory coordinates transformed to real word coordinates (i.e., UTM coordinates), they can be used for further transforming the point cloud coordinates to real word one.

The function Read_Trajectory (TrajFileName) is used to read all SLAM data. These data are then stored in an internal list called SlamTrajDataPoints which is a List of Trajectory Data Item (data as explained previously). Additionally, SLAM Trajectory data are stored in a high-performance list view (ListViewTrajectory) for visualization purposes.

4.4.5 Process

This is the main function of the LiDAR RTK software that combine RTK Trajectory data with SLAM Trajectory data. It performs several computations in different functions as follows:

- **Find Static Data** (FindStaticGPS): This function is used to determine the static positions within the RTK data which will be used in dividing the trajectory to small pieces (segments) in smoothing data function. Within the function FindStaticGPS, the static data are determined if distance between the sequence points is less than 5cm. The static data are stored in an internal list called TempStaticGPSData as follows: GPS Time From, GPS Time To, East, North, Height, Time Stamp From, Time Stamp To, Speed, Distance, Heading, Azimuth.
- **Interpolate RTK Trajectory** ("InterpolateRTK_Traj"): This is the first interpolation which is based on time interpolation to find the position (X, Y and Z) of RTK points within the original SLAM Trajectory. Two SLAM Trajectory points are selected for the interpolation which the RTK point located within. Then, the interpolation is conducted through the uses of "Interpolate_XYZ" function which performs linear interpolation of RTK position according to the Time Stamp difference of time RTK and the two selected SLAM Trajectory points. All SLAM Trajectory points as well as the interpolated points are stored in a list called "OutTraj_Interp_All". This will be then used in the next step of the processing which is smoothing.
- **Smooth data function** ("Smooth_Slam_AllData"): Noise in SLAM Trajectory data shown in wave shape. To construct a smooth trajectory, it was important to smooth these data to remove the wavy shape and any incorrect points. Smoothing process implemented as follows:
    - It starts by dividing the Trajectory into segments, each of which has four sequence lines. The first segment includes the first four lines, then the first line eliminated, and the following four lines are included in the second segment in sliding window method. This has been done using function "Find_Segments_StaticData" and results are stored in an internal list called Segments.
    - Then the SLAM Trajectory points within each segment are smoothed using a 6 degrees polynomial using the function "Smooth_data_PolynomialFit" which uses "MathNet.Numerics" library. The sequence points' distances are calculated, then utilized to create a distance data array called "Distdata" that stores the distances in accumulated form. The polynomial coefficients values are calculated using "MathNet.Numerics.Fit.Polynomial" function for the X data array and Distance data array ("Polynomial_X"), for the Y data array and Distance data array ("Polynomial_Y"), and for the Z data array and Distance data array ("Polynomial_Z"). Then using the function "Polynomial.Evaluate", the smoothed values of X, Y and Z coordinates of the SLAM Trajectory segments are calculated.

- Data of smoothed results are stored for only the first point, while other smoothed values are kept being averaged for the following smooth. In the following smoothing, the smooth values are averaged from current and previous smooth for the first point of the following smooth. This process repeated in all segments and the smoothed values of last segments are stored without averaging. The smoothed data are stored in a list called “OutSmooth_Data”.

- Interpolate RTK Trajectory (“InterpolateRTK_Traj”): This is the second interpolation which is also based on time interpolation to find the position (X, Y and Z) of RTK points within the smoothed SLAM Trajectory. Like the first interpolation, this interpolation aims to find the locations of the RTK positions within the smoothed SLAM trajectory for the final step. All SLAM Trajectory points as well as the interpolated points are stored in a list called “TrajectorySyncSmoothedDataPointsAll”. This will be then used in the next step of the processing which is transformation.
- **Transform SLAM Trajectory** (“Transform_Slam_Trajectory”): This is the last function that used to perform transformation for the SLAM Trajectory points to RTK Trajectory positions. The transformation function first compares the Time Stamp in both the SLAM Trajectory data and the RTK Trajectory data and store the matched ones in two list of integers called “RTK_IDs.Add” and “LiDAR_IDs.Add”. Then the first two RTK points and the corresponding SLAM Trajectory points are sent to “Find_Trans_Parameters” function for finding transformation parameters. The transformation parameters are calculated using 2D conformal transformation between the RTK East and North coordinates and the SLAM Trajectory X and Y coordinates. While the distance and elevation of RTK points and SLAM points are used to perform another 2D conformal transformation for the elevation. The function Transform2D_Conformal is used for the transformation parameters calculation.

  The calculated transformation parameters are then used to Transform all SLAM Trajectory points within the Time Stamp between the transformed points. This process is repeated for all points until the end of SLAM Trajectory data in sliding window method. The transformed data are stored in a list called “Tranformed_Data”.

4.4.6 Export

This function used to export vary number of internal lists and processed data results during different processing steps for further analysis. It makes 4 folders and export data as follows:

- **Object files folder** (“Obj_Files”): include all processed frames in .obj format.

- **PCD files folder** (“PCD_Files”): include all processed frames in Point Cloud Data (PCD) format. The PCD data comprises a set of 3D point coordinates that represent a point cloud. Point cloud data is commonly kept in PCD files, which can be used to display and analyze the scanned 3D world.
- **Static data folder** (“Static_Files”): it exports 3 ascii files: RTK_Data.txt, RTK_StaticData.txt and RTK_StaticSyncData.txt.
- **Trajectory data folder** (“Trajectory_Files”): it exports 6 ascii files: “Trajectory.txt”, “Traj_All_Data.txt”, “Smoothed_Trajectory.txt”, “Smoothed_Sync_Trajectory.txt”, “Smoothed_Sync_All_Trajectory.txt”, and the most important file that include final transformed Slam Trajectory file is “Transformed_Sync_All_Trajectory.txt”.

5 RESULTS, DISCUSSION AND EVALUATION

5.1 Results

As mentioned in 4.1 (Study Area), the outdoor test was carried out in Nablus area in Palestine close to the author's home. The test area was surveyed using Drone to provide as much as can with dense of points cloud that can be as a base for accuracy assessment of different tests. Phantom 4 RTK drone was used to collect 605 images at average altitude of about 77.5 meters above the natural ground to cover the study area. The images were collected in double grid: 12 strips East-West and another 13 strips North-East and all images were captured in RTK fixed position. Using GNSS, 15 Ground Control Points (GCPs) were collected using GNSS receiver precisely (GNSS report attached as Annex I. The images were post-processed using "Metashape Photoscan" version 1.8.3 software. The workflow for the post-processing was started by adding all images to the software in one chunk. Then, the position of cameras was converted to Palestine coordinates system to enable the use of the measured GCPs as well as to enable the use of the outputs for other assessments. After that, the images were aligned within "Metashape" software to create tie points. Then, 9 GCPs were used as control points while 6 points were used as check points. The RMSE of GCPs of control and check points is shown in Table 5.

Table 5: The RMSE of GCPs of control and check points

Count	X error (cm)	Y error (cm)	Z error (cm)	XY error (cm)	Total error (cm)	Type
9	1.23	0.75	1.22	1.44	1.88	Control
6	0.65	0.92	3.27	1.12	3.46	Check

As shown in Table 5, the total error of control point is less than 2cm. However, the total error of check points is found to be about 3.5cm and the maximum error were found in the Z component. A full report shows the results of different processing steps performed in "Metashape" software has been attached as Annex II. When analyzing the precision of the GNSS points, it was found that the Z precision is vary between 1.5cm to about 3cm. Furthermore, the maximum Z error in photogrammetry results of control points and check points was found to be about 2.3cm and 4.6cm respectively. The maximum Z error is in the check point was found in point number 7. Hence, the coordinates of this point were measured another two time to evaluate its elevation. The difference between the elevation of this point and the newly measured elevations were found to be 4cm and

2cm respectively. Hence, the error in point 7 could be results from a combination errors of GPS receiver and the photogrammetry model. After that, high accuracy dense point cloud, digital elevation model (DEM) with resolution of 4.12cm and orthophoto with resolution 2.0cm were generated.

For further analysis of the accuracy of the photogrammetry point cloud, about 300 GPS points were measured on the asphalt road. Using ArcMap 10.8, the elevation of these points was interpolated (linear interpolation) from the LiDAR dense cloud points using “add surface information” tool. The differences between the measured elevation of GPS points and the interpolated elevation were calculated and the RMSE of these differences were found to be equal to ±4.1cm. However, the maximum differences were found to be about 10cm in few locations.

5.1.1 Software Outcome

In addition, based on the comparison of the shape of points cloud in the developed LiDAR RTK and the Velodyne “Veloview”, they are very similar. That expresses the ability of the developed software to process LiDAR data accurately.

Based on the comparison of the developed LiDAR RTK software and Wireshark, there are differences between the Latitude and Longitude, and this is due to converting these values to decimal degrees in LiDAR RTK software. When converting these values to decimal degrees, they are matching. That expresses the ability of the developed software to process GPS data correctly.

- Based on the comparison of the shape point cloud in LiDAR RTK and “Veloview” software, they are very similar. That expresses the ability of the developed software to process LiDAR data accurately.
- GPS Time, the position quality, and speed are the same in LiDAR RTK and “Veloview” software. However, there are differences between the Latitude and Longitude, and this is due to converting these values to decimal degrees in LiDAR RTK software. When converting these values to decimal degrees, they are matching. That expresses the ability of the developed software to process GPS data correctly.

5.2 Accuracy Assessment of the RTK LiDAR

The outcomes of both indoor and outdoor tests are shown in this section.

5.2.1 Outdoor Palestine Test

To overcome the limitation of using NTRIP correction, a base station was set-up on the top of the author's home. Base and rover technique was used in the data collection. The developed device RTK LiDAR were mounted on a surveying pole and used for data collection of LiDAR and RTK data in Palestine Test area. The test was started from point 1 and nearly static data was collected to about 20 seconds after the RTK shows a fixed solution. Then, the data was continued to be recorded while walking to point 2. Again, nearly static data was collected on point 2 for about 20 seconds before moving to the next point. Then, similar data collection was carried out on all other points and the test was finished after closing the loop on the started point (point number 1). All measured data - LiDAR and RTK data- were transferred to computer and cleaning process was started to remove all unwanted data. The unwanted data was the LiDAR and RTK data collected before the RTK fixed solution was obtained. In total, 4260 LiDAR frame and 706 RTK positions were found.

The LiDAR data was post-processed using an open-source pre-compiled software called "VeloView" version 3.5 that contains advanced options to perform SLAM. The software was run to process all available LiDAR frames. The software was able to process all LiDAR frames and produce several outputs such as Trajectory data (result from SLAM process) and 3d data (Planner map and Edge map). Although the software was able to perform SLAM, but it was not considered the GPS positions available within the LiDAR PCAP file. Furthermore, a closure error of about 1.18m was in the SLAM trajectory see Figure 56.

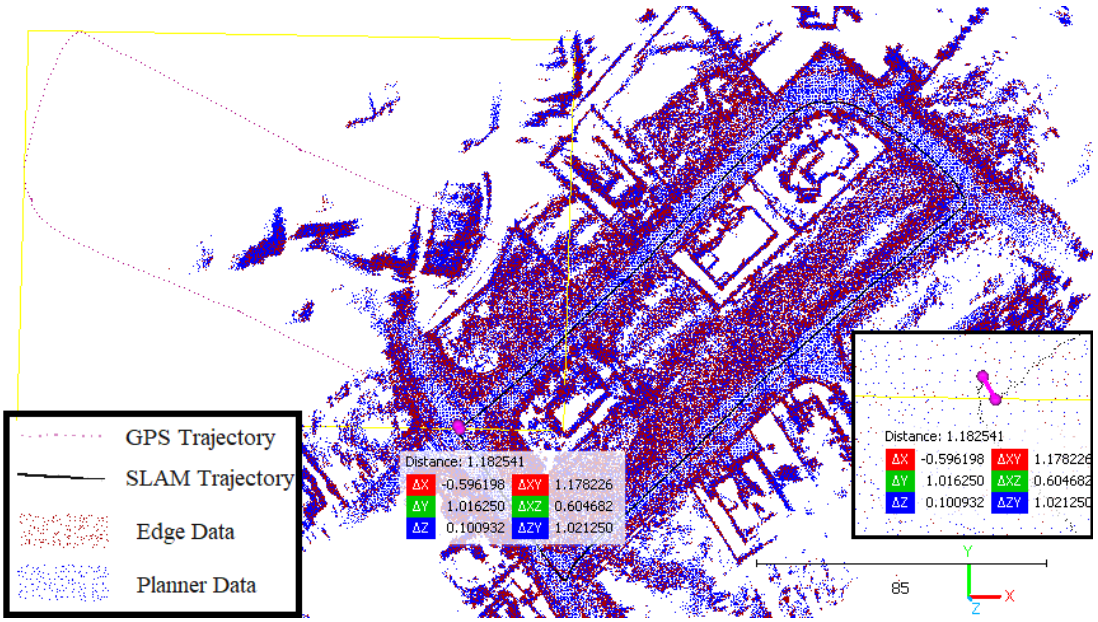


Figure 56: "Veloview" outputs of LiDAR post-processing data.

The closure error in the SLAM trajectory prevents the use of the output 3D points cloud generated by the "Veloview" software (Edge data and Planner data). Therefore, it is necessary to adjust the SLAM trajectory to eliminate the closure error and thus, to produce a precise 3D points cloud. The developed software (LiDAR RTK) was developed for the purpose of correcting the SLAM trajectory based on the precise RTK data. Hence, the collected LiDAR data (PCAP file) obtained from the developed device as well as the precise positional RTK data were used within the software to correct the SLAM trajectory. The processing steps that have been conducted in LiDAR RTK software for post-processing data of Palestine Test can be summarized as follows:

- **Open PCAP file**: when reading PCAP file, the software will find the following:
  - *Frames info*: find the start and end of packets and time stamp. It totals there were 4260 frame was found.
  - *GPS data*: the software will extract these data from positional packet recoded within the PCAP file and display information such as GPS time, latitude, longitude, quality, time stamp and speed. The total number of GPS data were 59049 data.
  - *LiDAR data*: for each frame, the software extract data as follows: azimuth, distance, intensity, laser id, time stamp and vertical angle. Also, it computes the coordinates (x, y and z) of all points within the current frame and plot them in the preview window as shown in Figure 57. The first frame has 18913 points. It is worth mentioning that it is possible to preview and plot data for each packet within the software.

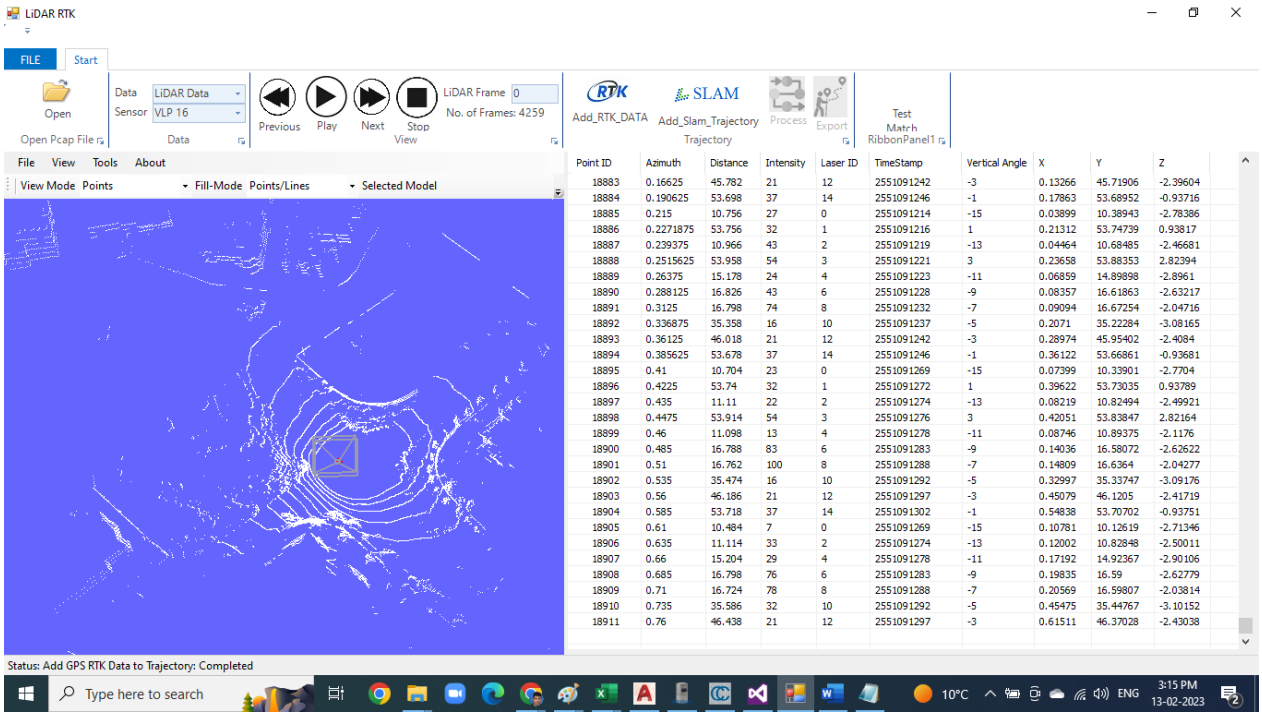


Figure 57: Data and plot of frame 0 points in LiDAR RTK software

- **Add RTK data**: when reading NMEA RTK file, the software will perform the following:
    - *RTK Trajectory*: the software will extract data from the GPRMC message as follows: GPS Time, Speed and Height. Also, it computes UTM East and North, and it computes the distance and azimuth between each two consecutive points. In addition, the software adds the heading information from GPS data extracted from the positional packet data recorded in the PCAP file. There is 706 points in the RTK trajectory file.
    - *GPS data*: the software will add the UTM East and North that computed during the process of adding RTK trajectory data. In addition, it will replace quality of GPS position to the RTK solution such as RTK Fixed. Most of positions were found that solution with RTK Fixed. However, Differential GNSS solution (DGNSS) solution was found at for 1 second and RTK Float was found at 8 seconds. These solutions will not be used during the modifying the SLAM trajectory.
- **Add Slam Trajectory**: when reading SLAM Trajectory file, the software will perform the following:
    - *Slam Trajectory*: the software imports trajectory data from the SLAM Trajectory file output from “Veloview” software. The data include data as follows: frame id, time stamp, rotation angles for each frame (Rx, Ry and Rz) and position coordinates (x, y and z) for each frame.
- **Process**: This function will perform the main processing steps that include: interpolation, smoothing and transformation to produce the final trajectory file.
- **Export**: this function will export different files that have been used through different processing steps. These files are such as all LiDAR frames in object and PCD formats, and trajectory files such as smoothed and transformed trajectory files. The different trajectory files are shown in Figure 58.

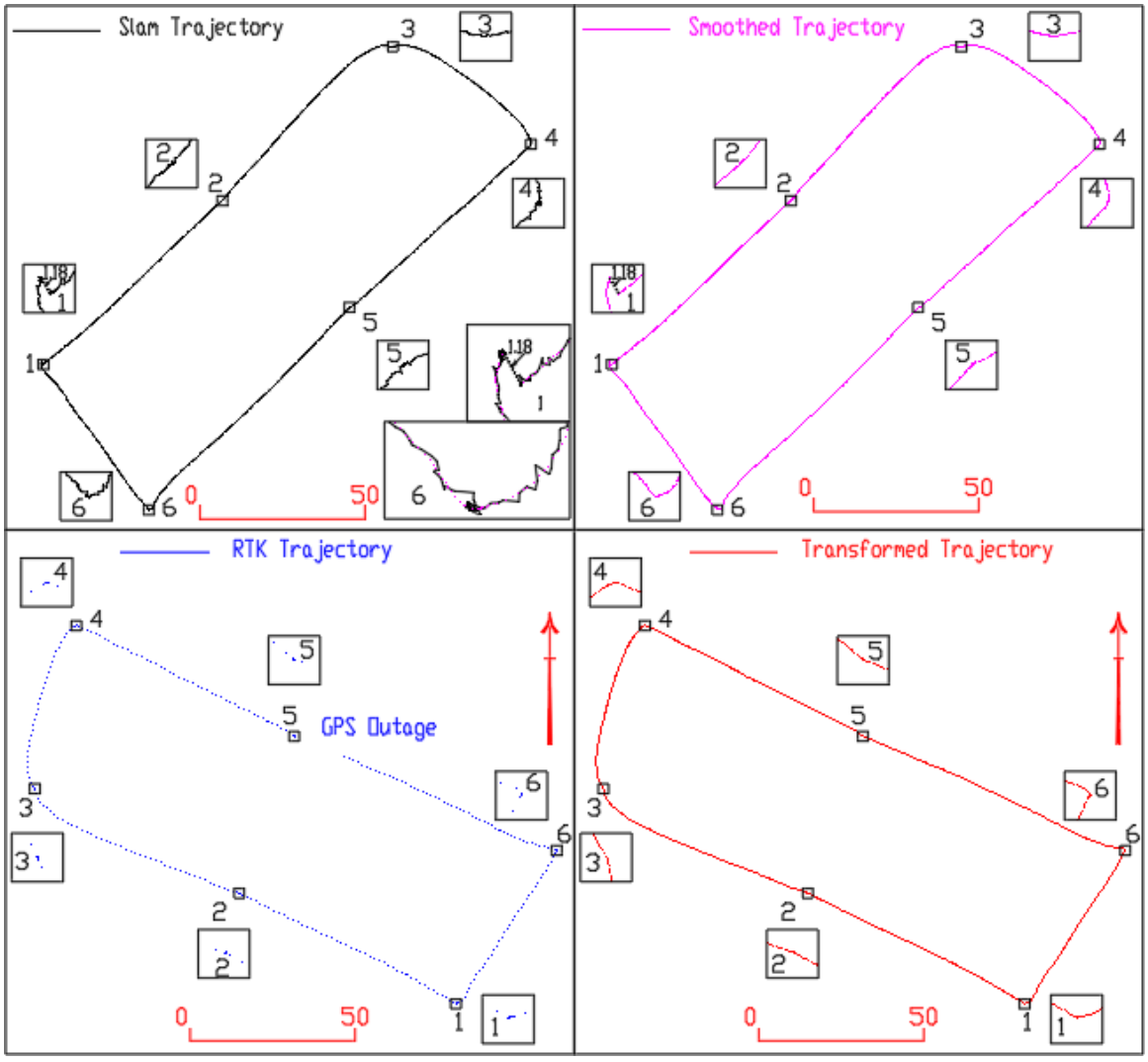


Figure 58: Slam Trajectory (Top left), Smoothed Trajectory (Top right), RTK Trajectory (Bottom left) and Transformed Trajectory (Bottom right)

A smooth step was required to make the SLAM trajectory reasonable because it is apparent upon closer inspection that it is fluctuating (see Figure 58 top left especially the zoom in areas 1 and 6). This is accurate given that the RTK Trajectory makes it clear that it is smooth. LiDAR RTK software combines the RTK trajectory with the smoothed trajectory to produce a transformed trajectory that also accounts for GPS outages. The fact that the developed RTK LiDAR device and LiDAR RTK software can be utilized to fill the gap left by a GPS outage may be regarded as one of the research's most important findings. To confirm this fact, a GPS outage in specific locations has been tested and the data was post-processed again using LiDAR RTK software.

5.3 Evaluation

5.3.1 Evaluation Criteria

Evaluation criteria are the standards for evaluating and assessing the research's results and its implications. To assess the accuracy of the Mobile LiDAR RTK, the location of well-defined natural features or landmarks will be measured precisely using different surveying techniques, including GPS, Total Station, or laser-Scanner. After that, LiDAR and GNSS data will be collected from the testing facility locations.

The LiDAR data will first be post-processed with open-source LiDAR software such as LiDAR View. Based solely on LiDAR observations, this software can perform SLAM algorithms to generate the user location file (trajectory file) and a map for the surveyed site (without GNSS data). At this stage, the relative accuracy of the well-defined targets will be evaluated. The software will be designed to read the LiDAR PCAP file and extract the GNSS observations from it. The software will also be used to generate a trajectory file based on GNSS data. A comparison of the SLAM location and the GNSS locations will be performed. The GNSS trajectory file will then be used within the LiDAR software to generate a georeferenced map of the surveyed site. Finally, as with the first assessment, the georeferenced map will be evaluated for the relative and absolute accuracy of well-defined targets within a pre-surveyed map.

5.3.2 Outdoor Test Area Accuracy Evaluation

As mentioned in 1.5 (Study Area), the outdoor test was carried out in Nablus area in Palestine close to the author’s home. The test area was surveyed using Drone to provide as much as can with dense of points cloud that can be as a base for accuracy assessment of different tests. Phantom 4 RTK drone was used to collect 605 images at average altitude of about 77.5 meters above the natural ground to cover the study area. The images were collected in double grid: 12 strips East-West and another 13 strips North-East and all images were captured in RTK fixed position. Using GNSS, 15 Ground Control Points (GCPs) were collected using GNSS receiver precisely (GNSS report attached as Annex I). The images were post-processed using “Metashape Photoscan” version 1.8.3 software. The workflow for the post-processing was started by adding all images to the software in one chunk. Then, the position of cameras was converted to Palestine coordinates system to enable the use of the measured GCPs as well as to enable the use of the outputs for other assessments. After that, the images were aligned within “Metashape” software

to create tie points. Then, 9 GCPs were used as control points while 6 points were used as check points. The RMSE of GCPs of control and check points is shown in Table 6.

Table 6: The RMSE of GCPs of control and check points

Count	X error (cm)	Y error (cm)	Z error (cm)	XY error (cm)	Total error (cm)	Type
9	1.23	0.75	1.22	1.44	1.88	Control
6	0.65	0.92	3.27	1.12	3.46	Check

As shown in Table 6, the total error of control point is less than 2cm. However, the total error of check points is found to be about 3.5cm and the maximum error were found in the Z component. A full report shows the results of different processing steps performed in “Metashape” software has been attached as Annex II. When analyzing the precision of the GNSS points, it was found that the Z precision is vary between 1.5cm to about 3cm. Furthermore, the maximum Z error in photogrammetry results of control points and check points was found to be about 2.3cm and 4.6cm respectively. The maximum Z error is in the check point was found in point number 7. Hence, the coordinates of this point were measured another two time to evaluate its elevation. The difference between the elevation of this point and the newly measured elevations were found to be 4cm and 2cm respectively. Hence, the error in point 7 could be results from a combination errors of GPS receiver and the photogrammetry model. After that, high accuracy dense point cloud, digital elevation model (DEM) with resolution of 4.12cm and orthophoto with resolution 2.0cm were generated.

For further analysis of the accuracy of the photogrammetry point cloud, about 300 GPS points were measured on the asphalt road. Using ArcMap 10.8, the elevation of these points was interpolated (linear interpolation) from the LiDAR dense cloud points using “add surface information” tool. The differences between the measured elevation of GPS points and the interpolated elevation were calculated (see Figure 59) and the RMSE of these differences were found to be equal to ±4.1cm. However, the maximum differences were found to be about 10cm in few locations.

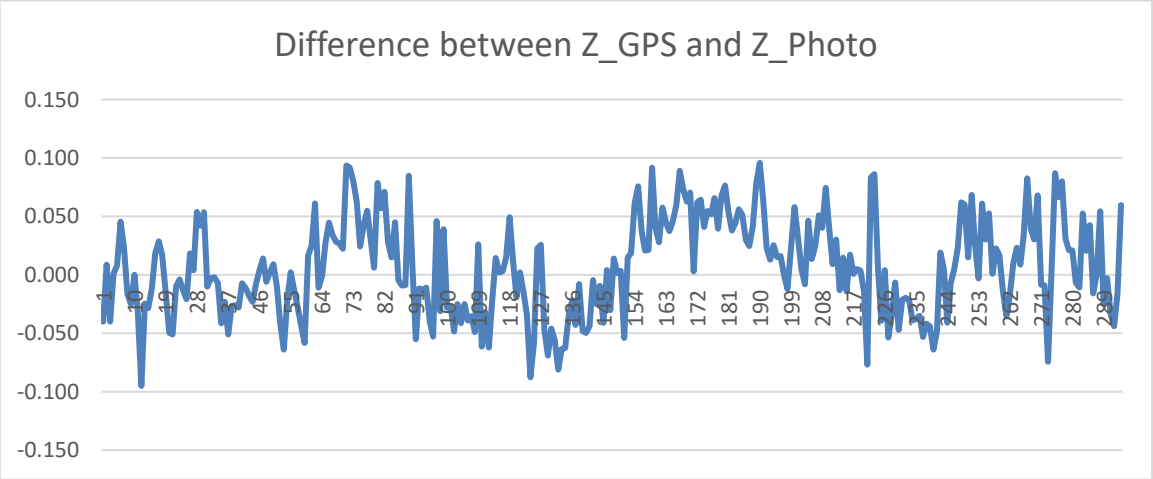


Figure 59: Difference between Elevations from GPS and Photogrammetry

The points with differences greater than 8cm are found in 35 locations and 15 points from them will be further verified. After remeasuring their elevations using GPS stake-out method, differences up to 10cm were found as shown in Figure 60. The main cause of the differences could be due to the use on Network correction as well as because the asphalt points have been not measured with high careful; the pole was not held exactly perpendicular on the points.

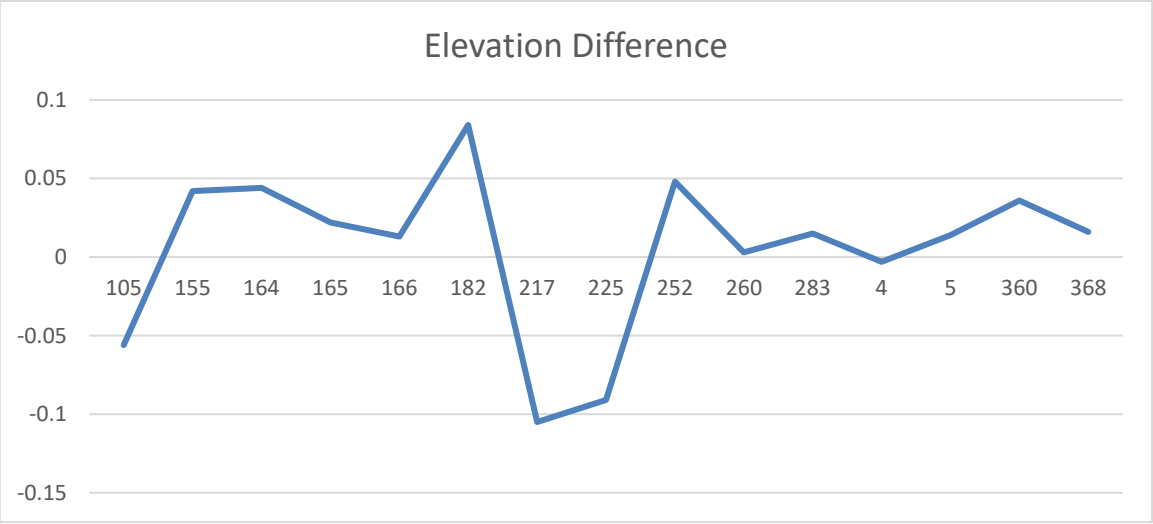


Figure 60: Difference between Elevations of same point in 14 locations GPS and Photogrammetry

For further analysis of the resultant points cloud obtained from photogrammetry model, total station (Topcon GTS 3005) has been used. This total station has the capability to measure objects with prism and with non-prism. First, the Total-Station was set-up on GCP number 1 and aimed

to GCP number 2. The coordinates of these points were fixed and the coordinates of GCP number 2 as well as GCPs 6, 7, 8, 13 and 14 were observed. The differences between GPS coordinates and Total-Station coordinated were computed and illustrated on Table 7.

Table 7: Differences between GPS and Total Station

Differences between GPS and Total Station in meter			
Point	**East**	**North**	**Height**
1	0.000	0.000	0.000
2	0.011	-0.006	-0.041
6	0.018	0.012	0.041
7	-0.007	-0.006	-0.043
8	0.012	-0.003	0.028
13	-0.015	-0.027	-0.008
14	-0.028	-0.024	-0.093
RMSE	**0.015**	**0.015**	**0.046**

The horizontal RMSE is 1.5cm while the vertical RMSE is 4.6cm with maximum error in the height of about 9cm. This confirms that the GPS measurements include errors similar to those errors found in the GPS stake-out results.

Another verification to the photogrammetry points cloud was carried out by utilizing about 160 Total-Station asphalt point around GCP number 1. Like the GPS asphalt points, the elevation of Total-Station asphalt point was interpolated from the photogrammetry points cloud using ArcMap software. The differences between the measured elevation of Total-Station points and the interpolated elevation were calculated (see Figure 61) and the RMSE of these differences were found to be equal to ±4.2cm. However, the maximum differences were found to be about 6cm.

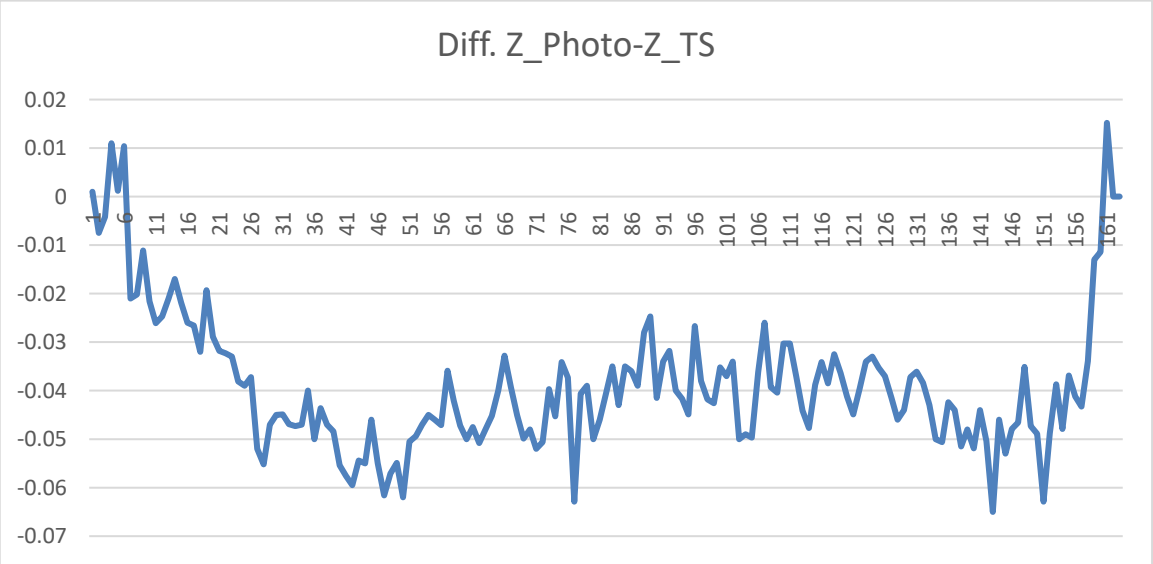


Figure 61: Difference between Elevations from Total-Station and Photogrammetry

Overall and based on the previous analysis, the accuracy of the points cloud obtained from photogrammetry model was found to be with RMSE of about ±4cm. Hence and under this condition, this model will be considered for further analysis of the LiDAR data.

5.3.2.1 Accuracy assessment of the VLP-16 LiDAR

Before assessment the accuracy of the collected LiDAR data, a brief analysis carried out for the VLP-16 LiDAR to evaluate the relative and absolute accuracy of such instrument in outdoor environment. To assess the relative accuracy, the differences between two sequence frames (frame 10 and frame 11) collected in static mode in outdoor environment was utilized. These frames were selected since there is no significant movement were noticed between them. It is worth mentioning that the developed LiDAR RTK was monumented on a surveying pole, therefore, it is difficult to hold the LiDAR exactly perpendicular on specific point as well as small movement could have existed. For analyzing the LiDAR output data, an open-source software called “CloudCompare” version 2.13 alpha was used. The color of the background of “CloudCompare” software was changed to white while the color of points cloud of frames 10 and 11 were modified to red and black respectively (Figure 62). Zoom in areas A and B was shown in Figure 62 to show the location of points cloud on these areas.

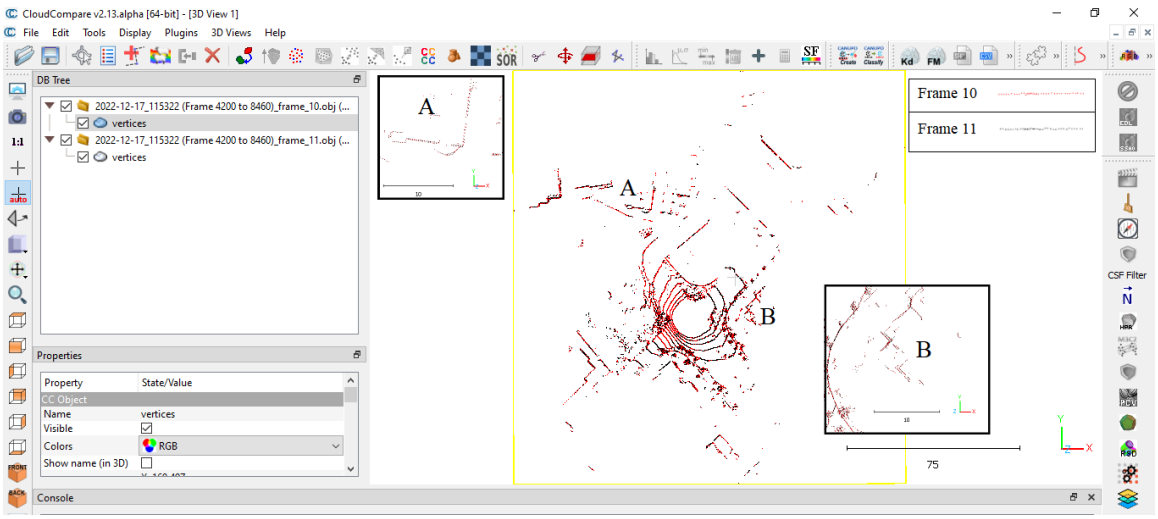


Figure 62: "CouldCompare" Software shows Point cloud of frame 10 and 11 with zoom in areas A and B

As shown from Figure 62, most point cloud are located on same locations on both frames 10 and 11. There is a small shift on the horizontal location of points cloud of frames since the LiDAR transmits different horizontal angles each frame. However, this will not cause any error in the location of the objects of different frames. For further analysis, the frames were converted to AutoCAD "dxf" files. The distance between the points cloud with similar pattern of both frames were measured in 12 areas for solid objects such as concrete wall and buildings and explained in Figure 63.

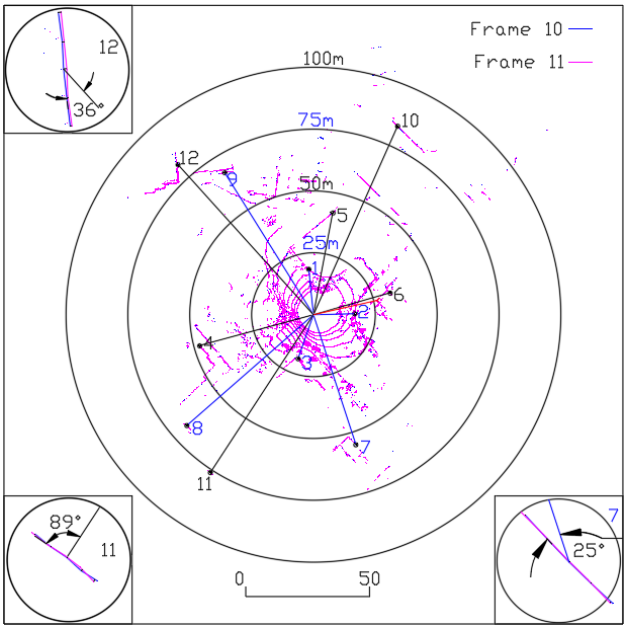


Figure 63: Relative Assessment, Point cloud of frames 10 and 11

On each area of the 12 locations, a polyline was drawn between 5 points of each frame and the perpendicular distance between these lines were calculated as shown in Table 8. For further analysis of the differences, the distance between the LiDAR and the object as well as the angle between this line and the surface are shown in Table 8 as well.

Table 8: Perpendicular distance between points in frames 10 and 11

Location	Perpendicular distance between points in frames 10 and 11 (m)					Average (m)	Distance From LiDAR (m)	Angle
1	0.013	0.019	0.015	0.027	0.021	**0.019**	18.435	43˚
2	0.002	0.003	0.001	0.006	0.005	**0.003**	16.662	86˚
3	0.007	0.009	0.007	0.007	0.006	**0.007**	18.828	52˚
4	0.010	0.018	0.024	0.018	0.000	**0.014**	47.728	59˚
5	0.021	0.023	0.016	0.002	0.046	**0.022**	41.926	40˚
6	0.039	0.016	0.001	0.011	0.043	**0.022**	32.110	70˚
7	0.010	0.005	0.009	0.004	0.002	**0.006**	55.368	25˚
8	0.014	0.030	0.008	0.008	0.015	**0.015**	67.987	71˚
9	0.038	0.008	0.006	0.027	0.048	**0.025**	67.681	63˚
10	0.007	0.008	0.010	**0.050**	0.014	**0.018**	83.411	65˚
11	0.011	0.001	0.005	0.021	0.033	**0.014**	76.196	89˚
12	0.047	0.032	0.054	0.019	0.098	**0.050**	81.707	36˚

As can be seen from Table 8, the average differences are found fluctuate from 0.003m to 0.05m. The large difference was found in location 12 which is 81.707m far away from LiDAR. The angle between the LiDAR and the object is about 36˚. However, the differences in other locations were found to be less than 0.02m. This indicates the high level of accuracy of this device in the sequence frames.

To assess the absolute accuracy of the measured LiDAR data, frame 11 were moved to point 1 since it was measured from that point. Then, the frame was rotated about 250˚ to match LiDAR points with features in the orthophoto that was obtained from drone. When computing the distance

between LiDAR points and features in locations 1,2,3 and 4 (in Figure 64) they were found to be equal to 0.50m, 0.44m, 0.24m and 0.28m respectively.

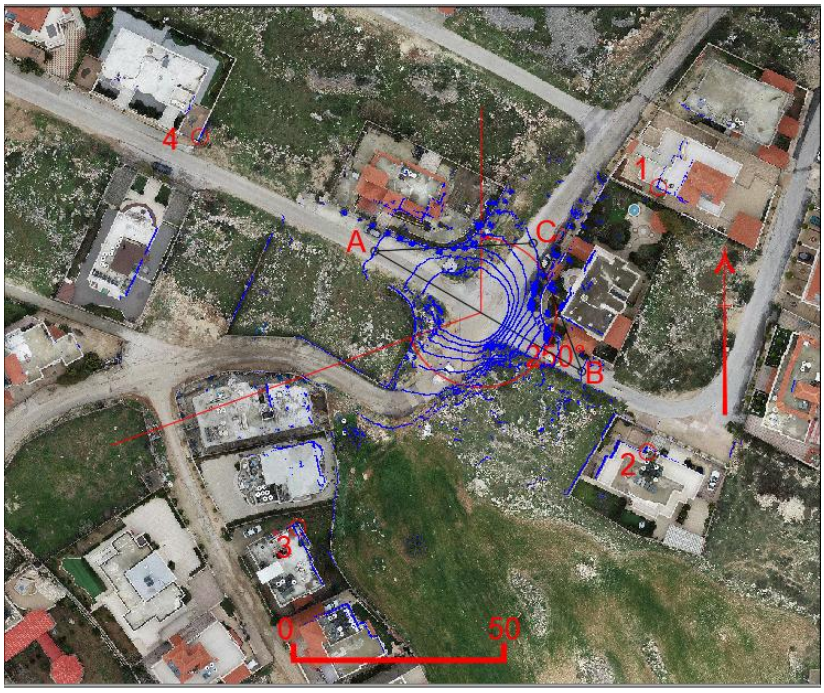


Figure 64: LiDAR Points of Frame 11 Superimposed on Orthophoto Map of Palestine Test

The main reason for the misalignment could be due to the differences of elevations between the LiDAR points cloud and the real world. To adjust the elevations of the LiDAR points cloud, 3 points were selected (A, B and C on Figure 64) on the asphalt road for this purpose. Using ArcMap, the elevation of these points was extracted from the photogrammetry point cloud file (las format file) (see Table 9). After that, the LiDAR points cloud aligned (3D alignment) within AutoCAD based on these elevations and the resultant points were moved again to the origin point 1. Again, the elevation of points A, B and C of modified location was extracted from the photogrammetry point cloud file (see Table 9) and a second 3D alignment was carried out. After that, the corner of buildings on location 1 and on location 3 was drawn from the intersected LiDAR points as well as from the orthophoto map. The horizontal distance between the points 1 and 3 were calculated and found to be 116.81m (LiDAR distance) and 116.74m (orthophoto distance). It is worth mentioning that the distance between 1 and 3 was measured by total station and found to be 116.85m. The elevation of points A, B and C was extracted again (see Table 9), and another 3D alignment was carried out for the third time to adjust the elevation of the LiDAR points.

Table 9: Palestine Test: The coordinates of points A, B and C during the alignment

Point	East	North	Elevation	Alignment No.
A	173207.665	179696.212	787.773	1
B	173256.913	179667.541	793.157	
C	173245.231	179698.109	789.018	
A	173207.447	179696.289	787.741	2
B	173256.970	179667.459	793.167	
C	173245.127	179698.086	789.018	
A	173207.564	179696.189	787.762	3
B	173257.084	179667.356	793.177	
C	173245.244	179697.985	789.023	

As can be seen from Table 9, the elevation of points A, B and C were changed within few centimeters (about 3cm) during different alignment. The small change of elevation could affect the overall accuracy of the LiDAR points. Hence, it is important to do the multiple alignment to ensure high level of accuracy for the LiDAR point.

After the step of alignment of LiDAR observation, it is important to assess the overall accuracy of the aligned LiDAR points. For this purpose, the LiDAR points that located on asphalt road have been extracted from the final alignment data (in total about 4800 asphalt points). The elevations of the extracted points were interpolated from photogrammetry points cloud using ArcMap software. The differences between the elevation of LiDAR points and the interpolated elevation from photogrammetry model were calculated (see Figure 65) and the RMSE of these differences were found to be equal to ±4.6cm. However, the maximum differences were found to be about 9cm in few locations.

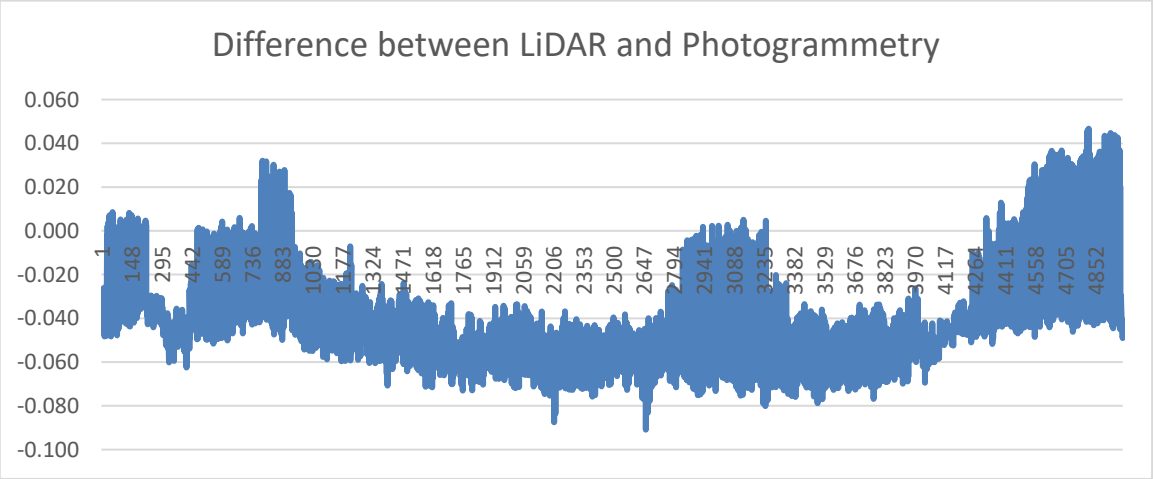


Figure 65: Palestine Test: Difference between Elevations from LiDAR and Photogrammetry

A three-dimensional plot of points cloud and LiDAR points was illustrated in Figure 66 using ArcScene 10.8. This Figure shows that all LiDAR points are fit correctly on the building's facades. In addition, when zoom in the LiDAR points still fit precisely to different features.

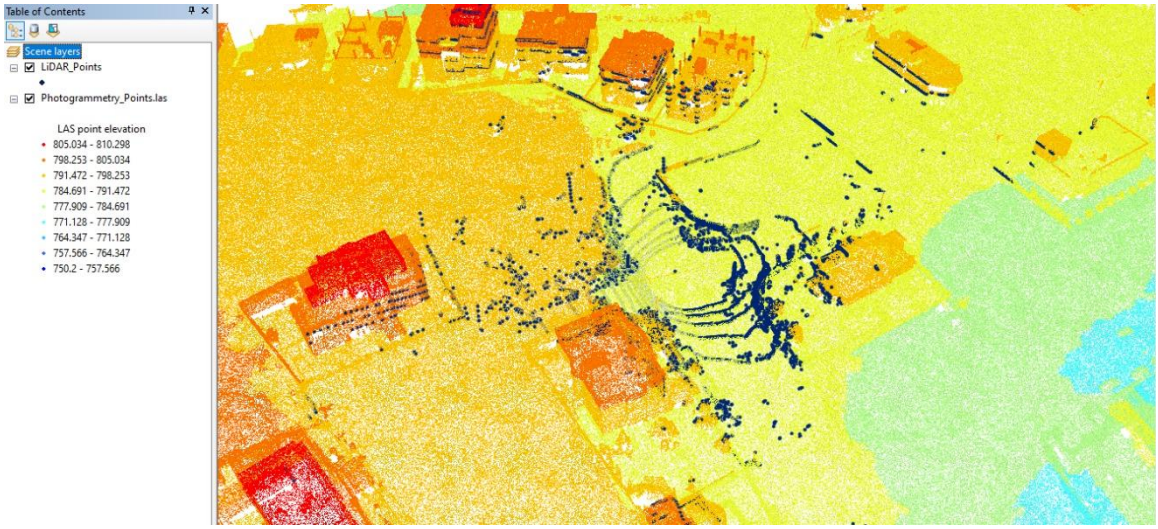


Figure 66: Palestine Test: A three-dimensional plot of LiDAR and Photogrammetry points

To assess the variations in heights accessible from the various technologies employed in this research (LiDAR, photogrammetry, and Total-Station), further analysis was conducted in 5 locations see Figure 67. The distance between LiDAR and Total-Station points was ranged from 2.6 cm to 8.6 cm.

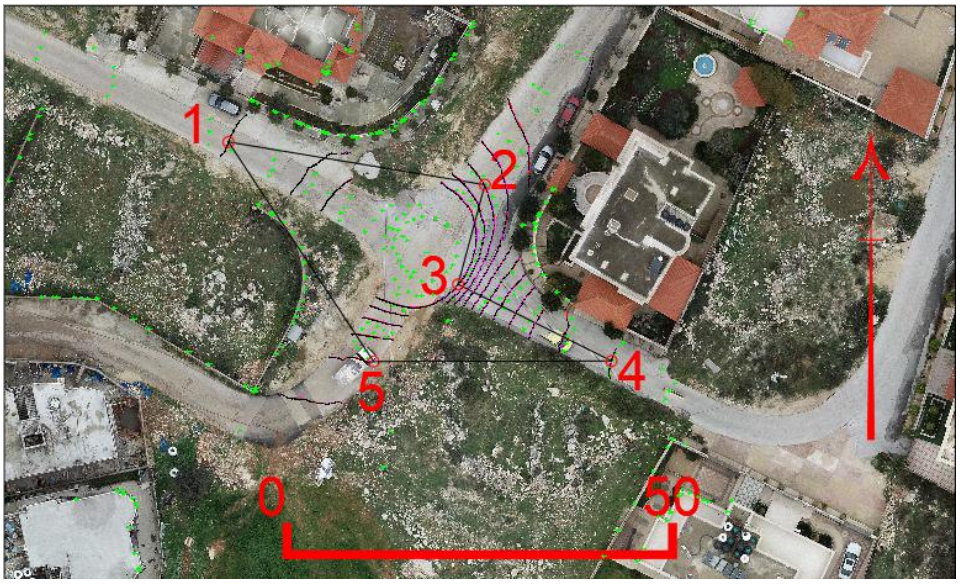


Figure 67: Palestine Test: Locations of points used for elevation comparison of LiDAR, photogrammetry, and Total-Station data.

Table 10 shows the elevation differences in 5 locations between LiDAR, photogrammetry, and Total-Station data, along with the distances between LiDAR and Total-Station points.

Table 10: Differences of elevation between LiDAR, photogrammetry, and Total-Station data with distances between LiDAR and Total-Station points

Point	LiDAR - Photogrammetry (cm)	LiDAR - TS (cm)	Distance between LiDAR and TS points (cm)
1	-0.20	-3.90	2.8
2	-2.90	0.70	8.6
3	-6.50	0.70	2.6
4	0.20	-5.10	6.7
5	-0.40	-5.60	6.7
RMSE	**3.19**	**3.84**	

As can be seen from Table 10, the RMSE of the elevation differences between LiDAR and Photogrammetry was found to be 3.19cm with errors from 0.2cm to 6.5cm. However, the RMSE of the elevation differences between LiDAR and Total-Station was found to be 3.84cm with errors

from 0.7cm to 5.6cm. Both results shows that the heights obtained from LiDAR data are accurate to few centimeters.

Overall, the analysis of the results from the comparison of LiDAR points with Total-Station and photogrammetry points demonstrates that the relative and absolute accuracy of LiDAR observations can reach a centimeter-level of precision. LiDAR technology is thus well reliable for use in surveying applications.

5.3.2.2 Creation of Three-Dimensional Maps

To proof the concept and to check the ability of creating 3d maps by utilizing the corrected trajectory, Palestine test data was post-processed between only two testing points (point 1 and point 2). Using “VeloView”, SLAM trajectory, Edge map and Planar map have been created. The SLAM trajectory was corrected and transformed to RTK data using LiDAR RTK software. Using AutoCAD and based on 3 trajectory points (testing points 1 and 2, and middle trajectory point), SLAM trajectory, Edge map and Planar map have been aligned to the corrected trajectory. The results of this transformation shown can be shown in Figure 68.

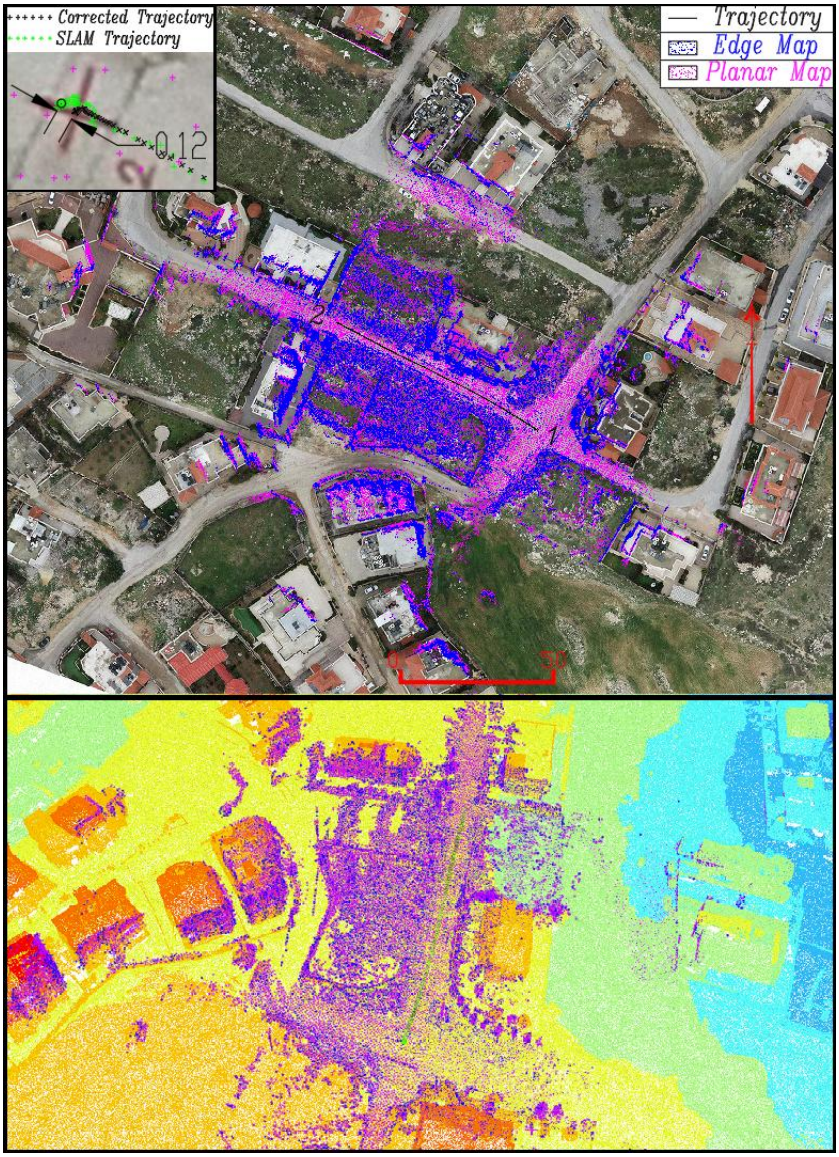


Figure 68: (Top) Plan map and (Bottom) 3D view of trajectory points, planar map, and edge map of Palestine test area

Figure 68 demonstrates that the 3d-created maps fit varied features (buildings, walls, and other features) satisfactorily on both the plan map and 3d view. Moreover, there is a 0.12-meter gap between the corrected trajectory and the SLAM trajectory. This may be the cause of the noisy point cloud observed on edge and planar maps. So, the SLAM process must be restored with the updated trajectory in mind. This capability is inapplicable within the VeloView software; hence, LiDAR RTK software could be enhanced in the future to address this issue. With the created MLRTK gadget and LiDAR RTK software, it is feasible to produce a precise 3D map. There is no necessity, in my opinion, to integrate a high-performance tactical IMU into the MLRTK gadget. This will keep the cost within the budgeted range for this study.

5.3.2.3 GPS Outage Filling evaluation

After completing the test and entering the GPS data, only GPS fixed points were used to build the RTK Trajectory. In addition, it was observed that a section of the RTK Trajectory lacks GPS reception, as depicted in Figure 69.

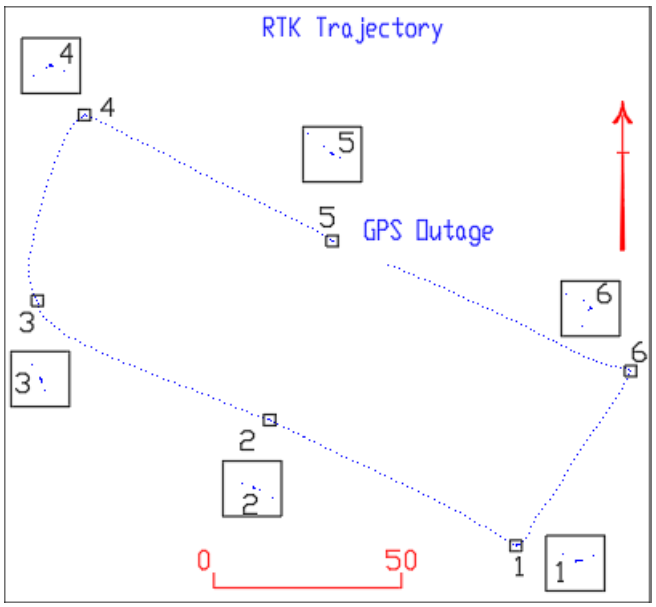


Figure 69: RTK trajectory GPS outage

When transforming the interpolated LiDAR trajectory, the GPS outage will be accounted for as demonstrated in Figure 70.

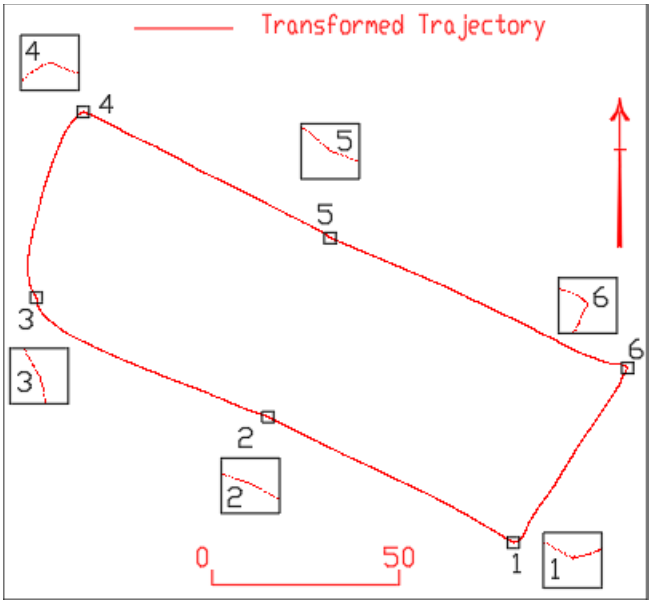


Figure 70: Transformed Trajectory

In order to evaluate this altered trajectory, a GPS point from the NMEA file was removed. In addition, this iteration occurred three times, with the first deletion of a GPS. In addition, 3 points were subtracted in the second iteration, and 5 points were subtracted in the final iteration. In addition, after reprocessing the interpolated trajectory, the result was compared to the initial RTK trajectory, as shown in table 11.

Table 11: Palestine Test: Interpolated and RTK trajectory coordinates.

GPS-Time	RTK			Interpolated		
	East	North	Height	East	North	Height
18:43:00	711529.7289	3566004.767	794.698	711529.7124	3566004.737	794.7232
18:43:00	711529.7289	3566004.767	794.698	711529.7173	3566004.746	794.74
18:43:01	711528.3062	3566005.53	794.563	711528.302	3566005.522	794.659
18:43:02	711526.9674	3566006.287	794.473	711526.9518	3566006.257	794.493
18:43:00	711529.7289	3566004.767	794.698	711529.7056	3566004.723	794.775
18:43:01	711528.3062	3566005.53	794.563	711528.2762	3566005.483	794.628
18:43:02	711526.9674	3566006.287	794.473	711526.9233	3566006.205	794.547
18:43:03	711525.6045	3566006.995	794.353	711525.5693	3566006.93	794.449
18:43:04	711524.2067	3566007.726	794.2	711524.1588	3566007.643	794.13

As stated in table 12, the differences between the points were then calculated.

Table 12: Palestine Test: Differences between RTK and Interpolated trajectory

Differences RTK-Interpolated			No. of Outage
D. East	D. North	D. Height	
0.017	0.030	-0.025	1
0.012	0.021	-0.042	3
0.004	0.008	**-0.096**	3
0.016	0.030	-0.020	3
0.023	0.044	-0.077	5
0.030	0.047	-0.065	5
0.044	**0.082**	-0.074	5
0.035	0.066	**-0.096**	5
0.048	0.083	0.070	5

In conclusion, the GPS outage can be addressed by the transformed interpolated trajectory as the discrepancies in accuracy were less than a few centimeters.

5.3.3 Indoor Test Area Accuracy Evaluation

Before assessing the accuracy of the obtained LiDAR data, a quick examination of the VLP-16 LiDAR's relative and absolute accuracy in both indoor and outdoor environments was conducted. The differences between two sequence frames (frame 340 and frame 341) captured in static mode in an outdoor environment were used to evaluate the relative accuracy. These frames were chosen because no substantial motion was seen between them. Notably, the generated LiDAR RTK was placed on a table; consequently, unlike the outdoor test, this test was completely static. For evaluating the LiDAR output data, the open-source software CloudCompare 2.13 alpha was utilized. The background color of CloudCompare software has been changed to white, and the point cloud colors of frames 340 and 341 have been switched to cyan and blue, respectively (Figure 71). Figure 71 depicts a magnification of regions A and B to illustrate the placement of the point cloud in these regions.

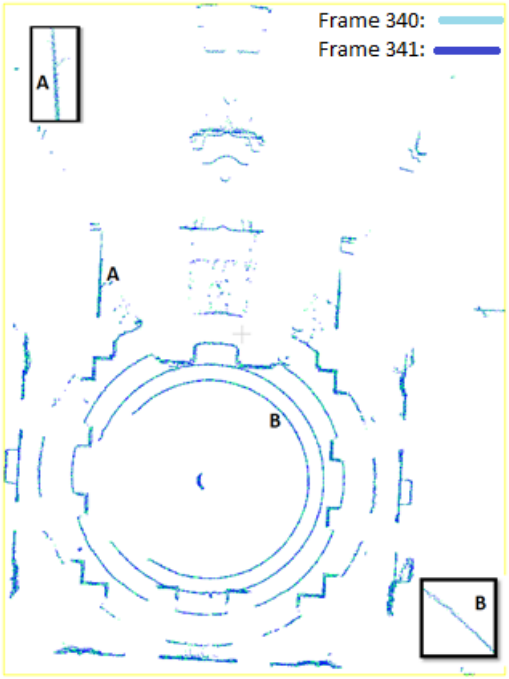


Figure 71: Germany Test: Point cloud of frame 340 and 341 with zoom in areas A and B

As seen in Figure 71, most of the point clouds on frames 340 and 341 are placed in the same areas. When the LiDAR transmits each frame at a different horizontal angle, there is a slight shift in the horizontal position of the point cloud. Yet, this will not affect the placement of items in different frames. For additional investigation, AutoCAD dxf files were generated from the frames.

As shown in Figure 72, the distance between point clouds with comparable patterns in both frames was assessed in 12 locations for solid objects such as concrete walls and columns.

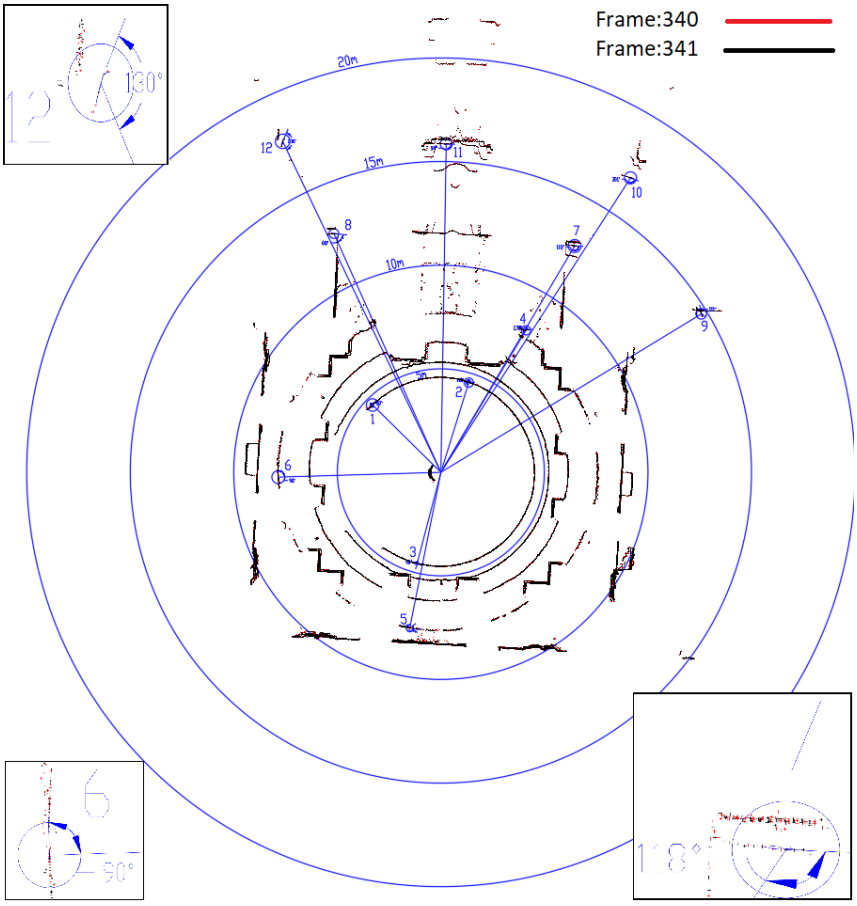


Figure 72: Germany Test: Points cloud of frames 340 and 341

For each of the 12 areas, a polyline was drawn between five points of each frame, and the perpendicular distance between these lines was computed according to Table 13. Table 13 also displays the distance between the LiDAR and the item, as well as the angle between this line and the surface, for additional study of the differences.

Table 13: Germany Test: Perpendicular distance between points in frames 340 and 341

Location	Perpendicular distance between points in frames 340 and 341 (m)				Average (m)		Distance From Li-DAR (m)	Angle
1	0.0118	0.0042	0.0084	0.0013	0.0029	**0.00572**	4.631	97º
2	0.0117	0.0182	0.0037	0.0058	0.0108	**0.01004**	4.534	110º
3	0.007	0.0088	0.0052	0.0016	0.0077	**0.00606**	4.458	90º
4	0.0224	0.0147	0.01	0.0034	0.005	**0.0111**	7.99	130º
5	0.003	0.0073	0.0057	0.0111	0.0099	**0.0074**	7.665	42º
6	0.0001	0	0	0	0	**0.00002**	7.856	90º
7	0.0198	0.0176	0.0082	0.0047	0.0083	**0.01172**	12.689	118º
8	0.0115	0.0026	0.0101	0.0128	0.0162	**0.01064**	12.628	60º
9	0.0073	0.011	0.0151	0.0146	0.0201	**0.01362**	14.72	111º
10	0.0043	0.0139	0.0118	0.0123	0.0152	**0.0115**	16.906	101º
11	0.0165	0.0013	0.0023	0.0034	0.0269	**0.01008**	15.849	90º
12	0.0122	0.0031	0.0109	0.0201	0.0306	**0.01538**	17.709	130º

As displayed in Table 13, the average differences range between 0.00002m and 0.01538m. The greatest difference was discovered at site 12, which is 17,709m from LiDAR. The angle between the LiDAR and the target is approximately 130 degrees. This means that this device's sequence frames are extremely precise.

To assess the absolute accuracy of the recorded LiDAR data, a tape was used to collect many measurements in various objects. With the LiDAR data, these measurements were then measured in AutoCAD as shown in figure 73.

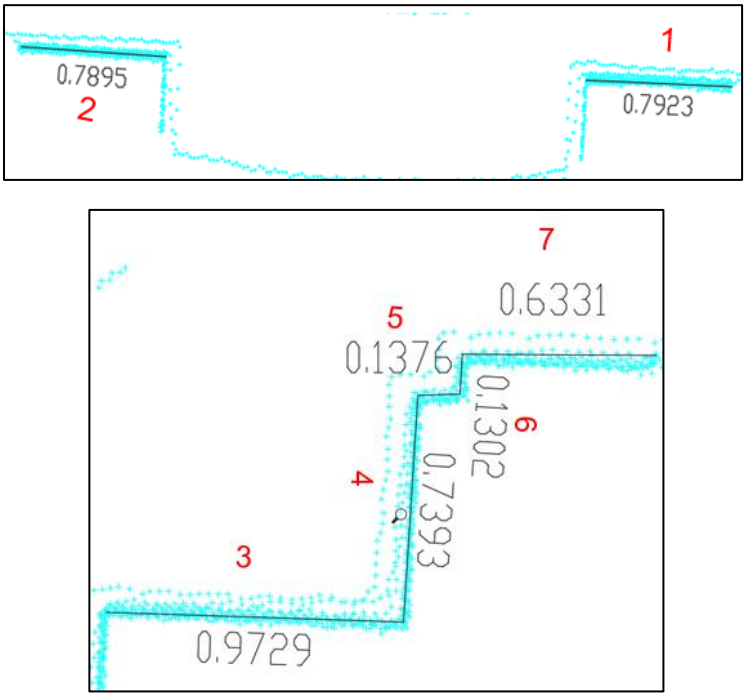


Figure 73: Germany Test: (Top) Features that were measured with LiDAR, (Bottom) Features that were measured with LiDAR.

The discrepancies between the LiDAR and tape measurements are calculated as stated in Table 14. In addition, the variances were less than 1 cm, which can be regarded to be an accurate measurement in an indoor environment as shown in Table 14.

Table 14: Germany Test: Differences between LiDAR and tape measurement.

Location	Tape measurement (m)	LiDAR measurement (m)	Differences
1	0.790	0.7923	-0.0023
2	0.790	0.7895	0.0005
3	0.980	0.9729	0.0071
4	0.740	0.7393	0.0007
5	0.138	0.1377	0.0003
6	0.130	0.1302	-0.0002
7	0.635	0.6331	0.0019

6 CONCLUSION AND FUTURE WORKS

6.1 Conclusion

Surveyors are searching for a low-cost device that may be utilized for mapping nearby areas and positioning. Two aims were proposed to accomplish these goals. The first aim was to cre-ate a low-cost device that cost less than €5,000, and the second goal was to create software that handles output data to accommodate the necessary applications.

The ability to combine and configure several sensors, including LiDAR, GNSS RTK receiver, Raspberry Pi microprocessor, and other sensors that can work together to collect LiDAR and RTK data, was made possible after intensive research. The created device, which had a total price of roughly €3,500, was lightweight (weighing about 1.825kg). The MLRTK gadget is relatively simple to use for data collection. Turn on the power, launch the NTRIP client app on your phone, pair it with a GNSS receiver using Bluetooth, wait for the RTK fixed solution, launch LiDAR logging on your Raspberry Pi, and begin gathering data. When collecting data, it is preferable to employ the Stop & Go with closed loop technique. While the RTK data can be sent through Bluetooth from the mobile phone, the gathered LiDAR data (PCAP file) can be downloaded from the Raspberry Pi.

In this study, a new program called LiDAR RTK was created using Microsoft Studio (Visual Basic .NET) and based on several libraries. When reading the binary PCAP file, the positional and LiDAR data can both be extracted. Following that, the LiDAR data can be visualized and saved in PCD and OBJ formats. Additionally, this software can parse RTK NMEA data to update time-synchronized positional data and generate RTK trajectory. Moreover, through a series of interpolation, smoothing, and transformation phases which performs in sliding window method, time stamp SLAM trajectory data can be processed alongside RTK trajectory data to provide corrected trajectory data. Since the SLAM trajectory and the point cloud data are in the same coordinate system, the corrected trajectory can then be utilized to transform point cloud and hence to produce a transformed 3D map.

Using the developed MLRTK device, both LiDAR and RTK data were successfully collected in the Palestine test area. There was, however, a GPS outage in one area. The LiDAR data was then post-processed using an open-source “VeloView” software to produce a time stamp SLAM trajectory and 3d map. However, a closure error of about 1.18m was found on the start-ing point. Using, LiDAR RTK software, it was possible to correct the SLAM trajectory, to eliminate the

closure error, to produce a smoothed trajectory and to fill the GPS outage gap. Based on the modified trajectory, additional adjustments should be made to the point cloud produced by the SLAM process. Nevertheless, “VeloView “does not currently support this, thus specialized software should be developed to accomplish this. It is important to note that one of the research's most important findings is the ability to use the MLRTK device and software created within this research to bridge the GNSS outage gap.

To proof the concept and to check the ability of creating a 3d map by utilizing the corrected trajectory, Palestine test data was post-processed between only two testing points (point 1 and point 2). The results shows that there is a 0.12-meter gap between the corrected trajectory and the SLAM trajectory which causes the noisy point cloud observed on edge and planar maps. Moreover, with the created MLRTK gadget and LiDAR RTK software, it is feasible to produce a precise 3D map without the need of a high-performance tactical IMU.

The evaluation of the LiDAR data collected in both outdoor and indoor environments shown centimeter level of accuracy obtained in both relative and absolute accuracy assessment tests. The distance between sequence frames during the outdoor test did not exceed 5 cm. In addition, variation on elevation between the LiDAR frame and photogrammetry measurements did not exceed 6.5 cm, while disparities between the LiDAR and the nearest TS measurement did not exceed 5.6 cm. For the indoor test, the differences between sequence frames did not exceed 1.5 centimeters, and the discrepancies between the LiDAR and tape measurements were accurate to centimeter level of accuracy.

To summarize, the outcome of this research was the development both a low-cost, lightweight hardware device (MLRTK) and an easy-to-use software application (LiDAR RTK). Collecting data using MLRTK device and following the processing procedure that based on LiDAR RTK software, surveyors can determine their location and map the surrounding area. More important, the developed techniques can be utilized not only in both indoor and outdoor environments, but even in the absence of GPS to provide a transformed trajectory with a smoother path.

6.2 Future work

The work in this thesis focuses on two major fields of research: the hardware integration and software development.

For the first part of the research, after extensive research and configurations, it was possible to develop the MLRTK device that successfully used by surveyors for data collection. However, one improvement could be through the upgrading this device to be able to collect RTK GNSS data rather than logging the data using an external mobile phone.

For the second part of this research, it was necessary to develop a software that can deals with different types of data include binary PCAP data, NMEA data and SLAM trajectory data. The developed LiDAR RTK software was able to produce excellent results of corrected trajectory which have been through a series of interpolation, smoothing and transformation. Although the software can read the binary PCAP data, it is not currently possible to create a SLAM trajectory. For future work, it is recommended of implementing a SLAM within the LiDAR RTK software that uses the GPS Trajectory into its construction. Also, implement a depth camera to color the point cloud. Lastly, keeping the project current with continuous, quick technological developments.

REFERENCES

[1] "Stuttgart | Germany, Map, History, & Points of Interest | Britannica." https://www.britannica.com/place/Stuttgart-Germany (accessed Feb. 23, 2023).

[2] J. Ryan Kidd and J. Ryan, "Performance Evaluation of the Velodyne VLP-16 System for Surface Feature Surveying Recommended Citation." [Online]. Available: https://scholars.unh.edu/thesis/1116

[3] A. Kukko, H. Kaartinen, J. Hyyppä, and Y. Chen, "Multiplatform mobile laser scanning: Usability and performance," *Sensors (Switzerland)*, vol. 12, no. 9. pp. 11712–11733, Sep. 2012. doi: 10.3390/s120911712.

[4] L. Chang, X. Niu, and T. Liu, "Gnss/imu/odo/lidar-slam integrated navigation system using imu/odo pre-integration," *Sensors (Switzerland)*, vol. 20, no. 17, pp. 1–18, Sep. 2020, doi: 10.3390/s20174702.

[5] P. Yu, M. Wang, and H. Chen, "Integration and evaluation of SLAM-based backpack mobile mapping system," in *E3S Web of Conferences*, Nov. 2020, vol. 206. doi: 10.1051/e3sconf/202020603014.

[6] M. R. James and J. N. Quinton, "Ultra-rapid topographic surveying for complex environments: The hand-held mobile laser scanner (HMLS)," *Earth Surf Process Landf*, vol. 39, no. 1, pp. 138–142, 2014, doi: 10.1002/esp.3489.

[7] M. Holopainen *et al.*, "Tree mapping using airborne, terrestrial and mobile laser scanning - A case study in a heterogeneous urban forest," *Urban For Urban Green*, vol. 12, no. 4, pp. 546–553, 2013, doi: 10.1016/j.ufug.2013.06.002.

[8] "Faro Photon 120 Laser Scanner Set for sale – 3D Laser Scanner manufacturer from china (101379801)." https://borneosurveying.sell.everychina.com/p-101379801-faro-photon-120-laser-scanner-set.html (accessed Sep. 24, 2022).

[9] "ZEB Horizon Solution including GeoSLAM HUB Software & 1 YR GeoSLAM care." https://allterracentral.com/zeb-horizon-solution-including-geoslam-hub-software-1-yr-geoslam-care.html (accessed Sep. 21, 2022).

[10] "Paracosm PX-80 review - handheld LiDAR 3D scanner." https://www.aniwaa.com/product/3d-scanners/paracosm-px-80/ (accessed Sep. 21, 2022).

[11] "Greenvalley LiGrip | Lowest price online | Global GPS Systems." https://globalgps-systems.com/greenvalley-ligrip/ (accessed Sep. 21, 2022).

[12] "VLP-16 User Manual," 2019. [Online]. Available: http://www.velodynelidar.com/downloads.html#firmware.

[13] "Demonstration of multiple-return LiDAR technology. The height of an... | Download Scientific Diagram." https://www.researchgate.net/figure/Demonstration-of-multiple-return-LiDAR-technology-The-height-of-an-object-can-be_fig1_236325446 (accessed Feb. 17, 2023).

[14] "Time of flight - Wikipedia." https://en.wikipedia.org/wiki/Time_of_flight (accessed Feb. 24, 2023).

[15] "The difference in phase of two waves is known as the phase shift | Download Scientific Diagram." https://www.researchgate.net/figure/The-difference-in-phase-of-two-waves-is-known-as-the-phase-shift_fig1_266234756 (accessed Feb. 17, 2023).

[16] "Wide Field of View with Enhanced Point Density and Range." [Online]. Available: www.velodynelidar.com

[17] N. Li *et al.*, "A Progress Review on Solid-State LiDAR and Nanophotonics-Based LiDAR Sensors," *Laser and Photonics Reviews*. John Wiley and Sons Inc, Nov. 01, 2022. doi: 10.1002/lpor.202100511.

[18] "4: (a) Flash illumination-based LiDAR and (b) beam scanning-based LiDAR. | Download Scientific Diagram." https://www.researchgate.net/figure/a-Flash-illumination-based-LiDAR-and-b-beam-scanning-based-LiDAR_fig6_336070619 (accessed Feb. 03, 2023).

[19] S. Royo and M. Ballesta-Garcia, "An overview of lidar imaging systems for autonomous vehicles," *Applied Sciences (Switzerland)*, vol. 9, no. 19, Oct. 2019, doi: 10.3390/app9194093.

[20] "(a) The structured light camera design uses an 1D MEMS mirror and... | Download Scientific Diagram." https://www.researchgate.net/figure/a-The-structured-light-camera-design-uses-an-1D-MEMS-mirror-and-diffused-laser-52-b_fig1_340971588 (accessed Feb. 03, 2023).

[21] "Schematic showing the working principle of a typical LiDAR system based... | Download Scientific Diagram." https://www.researchgate.net/figure/Schematic-showing-the-

working-principle-of-a-typical-LiDAR-system-based-on-MEMS-OPA_fig2_351685963 (accessed Feb. 03, 2023).

[22] "VELABIT SOLID-STATE LiDAR SENSOR." https://leodrive.ai/products/velabit-solid-state-lidar-sensor (accessed Feb. 20, 2023).

[23] "Trimble X7 Laser Scanner | What You Need to Know | BuildingPoint Mid-America." https://www.bpmidamerica.com/blog/introducing-trimble-x7-laser-scanner/ (accessed Feb. 17, 2023).

[24] "https://en.wikipedia.org/wiki/List_of_SLAM_methods."

[25] "ICP Registration — Open3D latest (664eff5) documentation." http://www.open3d.org/docs/latest/tutorial/Basic/icp_registration.html (accessed Feb. 17, 2023).

[26] A. W. Fitzgibbon, "Robust Registration of 2D and 3D Point Sets."

[27] D. Spinczyk and M. Bas, "Anisotropic non-rigid Iterative Closest Point Algorithm for respiratory motion abdominal surface matching," *Biomed Eng Online*, vol. 18, no. 1, Mar. 2019, doi: 10.1186/s12938-019-0643-4.

[28] "Universal Transverse Mercator coordinate system - Wikipedia." https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system (accessed Feb. 05, 2023).

[29] "Universal Transverse Mercator (UTM) coordinate system is a standard set... | Download Scientific Diagram." https://www.researchgate.net/figure/Universal-Transverse-Mercator-UTM-coordinate-system-is-a-standard-set-of-map_fig3_342330834 (accessed Feb. 05, 2023).

[30] "Local coordinate systems | Moldflow Insight | Autodesk Knowledge Network." https://knowledge.autodesk.com/support/moldflow-insight/learn-explore/caas/CloudHelp/cloudhelp/2016/ENU/MoldflowInsight/files/GUID-D977E3B1-0789-413B-AF77-40099DF6E285-htm.html (accessed Feb. 18, 2023).

[31] "FEFLOW 7.3 Documentation - Coordinate Systems." http://www.feflow.info/html/help73/feflow/14_References/coordinate_systems.html (accessed Feb. 18, 2023).

[32] "3D lidar SLAM: The Basics | Kudan global." https://www.kudan.io/blog/3d-lidar-slam-the-basics/ (accessed Feb. 18, 2023).

[33] "https://receiverhelp.trimble.com/alloy-gnss/en-us/NMEA-0183messages_RMC.html."

[34] "https://receiverhelp.trimble.com/alloy-gnss/en-us/NMEA-0183messages_GGA.html."

[35] J. F. Kurose and K. W. Ross, *Computer networking : a top-down approach*. Pearson, 2013.

[36] "3D lidar SLAM: The Basics | Kudan global." https://www.kudan.io/blog/3d-lidar-slam-the-basics/ (accessed Feb. 19, 2023).

[37] "Alberding GmbH - Software >> GNSS Data Management Software (Ntrip) >> Overview." https://www.alberding.eu/de/ntripcaster.html (accessed Feb. 19, 2023).

[38] "VRS concept representation. | Download Scientific Diagram." https://www.researchgate.net/figure/VRS-concept-representation_fig1_334280090 (accessed Feb. 19, 2023).

[39] "Horizon_Spec_Sheet".

[40] "VeloView: Lidar SLAM capabilities." https://www.kitware.com/veloview-lidar-slam-capabilities/ (accessed Feb. 20, 2023).

[41] "Releases · LidarView / LidarView · GitLab." https://gitlab.kitware.com/LidarView/lidarview/-/releases (accessed Feb. 20, 2023).

[42] "paraview_wrapping/Plugin/doc/How_to_SLAM_with_LidarView.md · master · KEU-ComputerVision / Slam · GitLab." https://gitlab.kitware.com/keu-computervision/slam/-/blob/master/paraview_wrapping/Plugin/doc/How_to_SLAM_with_LidarView.md (accessed Feb. 20, 2023).

[43] "Fast Point Cloud Viewer with C# and OpenGL - CodeProject." https://www.codeproject.com/Articles/839389/Fast-Point-Cloud-Viewer-with-Csharp-and-OpenGL (accessed Feb. 21, 2023).

[44] "Wavefront .obj file - Wikipedia." https://en.wikipedia.org/wiki/Wavefront_.obj_file (accessed Feb. 21, 2023).

[45] "Displaying Large Quantities of Data in Windows Controls - CodeProject." https://www.codeproject.com/Articles/37641/Displaying-Large-Quantities-of-Data-in-Windows-Con (accessed Feb. 21, 2023).

[46] "Easily Add a Ribbon into a WinForms Application - CodeProject." https://www.codeproject.com/Articles/364272/Easily-Add-a-Ribbon-into-a-WinForms-Application (accessed Feb. 21, 2023).

[47] "Developer Guide." https://coordinatesharp.com/DeveloperGuide (accessed Feb. 21, 2023).

[48] "GitHub - mathnet/mathnet-numerics: Math.NET Numerics." https://github.com/mathnet/mathnet-numerics (accessed Feb. 21, 2023).

ANNEXES

I. GNSS Report

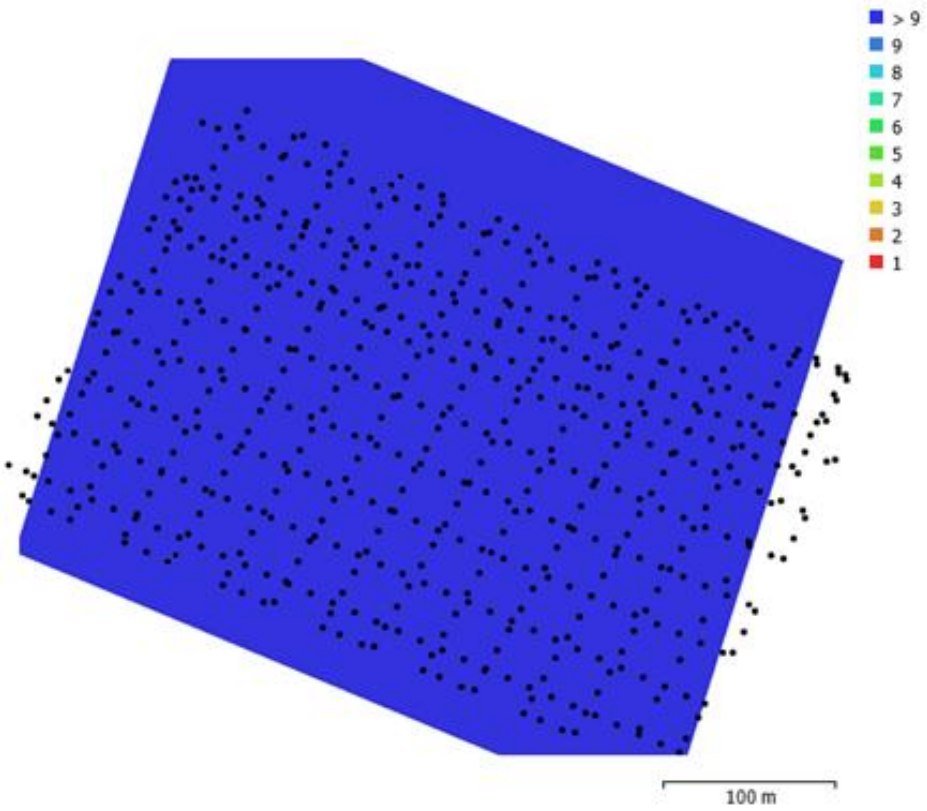
The coordinates of GCPs that were used in Palestine test area are listed in Table I.1.

Table 15: GNSS report of GCPs in Palestine Testing Area

ID	East	North	Height	H. Precision	V. Precision	GPS Sats.	GLONASS Sats.
1	173232.889	179681.622	789.978	0.009	0.015	8	4
2	173167.904	179716.525	785.738	0.009	0.016	8	4
3	173106.648	179749.655	785.277	0.015	0.023	4	5
4	173120.337	179798.969	778.15	0.01	0.017	7	4
5	173185.551	179763.895	776.851	0.01	0.017	7	4
6	173264.345	179727.536	786.308	0.01	0.018	7	4
7	173189.794	179708.277	786.426	0.009	0.015	8	4
7-2.	173189.812	179708.28	786.386	0.012	0.013	9	3
7-3.	173189.827	179708.291	786.406	0.068	0.057	7	0
8	173137.283	179727.234	785.702	0.012	0.018	6	5
9	173147.257	179786.21	777.808	0.01	0.017	7	4
10	173219.174	179752.889	778.748	0.009	0.015	7	5
11	173288.19	179659.413	795.563	0.01	0.018	7	4
12	173303.985	179704.682	791.9	0.011	0.024	6	4
13	173208.778	179658.427	790.562	0.01	0.019	7	4
14	173153.821	179672.503	789.275	0.011	0.029	6	4
15	173074.295	179703.542	790.542	0.009	0.018	7	4

II. Agisoft Metashape Report

Figure II.1 shows camera locations and image overlap. As can be seen from the figure below, all area were covered by 9 images.



Number of images:	605	Camera stations:	605
Flying altitude:	77.5 m	Tie points:	134,399
Ground resolution:	2.06 cm/pix	Projections:	1,676,254
Coverage area:	0.11 km²	Reprojection error:	0.476 pix

Figure 74: Camera locations and image overlap.

The camera model and its properties are shown in Table II.1.

Table 16: Cameras.

Camera Model	Resolution	Focal Length	Pixel Size	Precalibrated
FC6310R (8.8mm)	5472 x 3648	8.8 mm	2.41 x 2.41 µm	Yes

Ground Control Points

Table II.2 demonstrates the GCP and the error in the 3D coordinates.

Table 17: Control points.

Label	**X error (cm)**	**Y error (cm)**	**Z error (cm)**	**Total (cm)**	**Image (pix)**
2	1.367	-0.108	-2.278	2.659	0.322 (8)
3	1.935	-0.065	-0.095	1.939	0.166 (8)
4	-1.405	-1.315	0.734	2.060	0.216 (8)
5	-0.648	0.015	0.780	1.015	0.247 (8)
11	-1.313	-0.422	1.040	1.728	0.291 (8)
12	1.785	1.044	-0.351	2.097	0.188 (8)
13	0.181	-0.533	0.488	0.745	0.209 (8)
14	-0.742	-0.076	-2.216	2.338	0.185 (8)
15	-0.181	1.311	0.815	1.555	0.198 (8)
Total	**1.2267**	**0.747**	**1.218**	**1.883**	**0.230**

The error in the 3D coordinates in the check points are shown in Table II.3.

Table 18: Check points.

Label	**X error (cm)**	**Y error (cm)**	**Z error (cm)**	**Total (cm)**	**Image (pix)**
1	-0.255	0.282	-3.0259	3.049	0.168 (8)
6	-0.183	0.621	-1.72883	1.846	0.183 (8)
7	-0.098	-0.142	-4.56757	4.570	0.183 (8)
8	1.359	-0.809	3.51839	3.857	0.203 (8)
9	0.405	-0.265	3.90862	3.938	0.166 (8)
10	-0.639	-1.961	1.91824	2.817	0.178 (8)
Total	**0.649**	**0.918**	**3.274**	**3.462**	**0.181**