Progress Report 1

Kasra Ekrad*, Andreas Johansson†, Walter Lagerhäll‡, Sebastian Leclerc§,
Elon Pettersson¶, Erik Rågberger‖, Sharifeh Yaghoobi**
School of Innovation, Design and Engineering
Mälardalens University, Västerås, Sweden
Email: *ked22001, †ajn19017, ‡wll19001, §sic19001, ¶epn19005, ‖err19002, **syi22001@student.mdu.se

I. Project Information

Project Name	Safe Trustworthy Autonomous Delivery System
Stakeholders	Mälardalen University
Project Manager	Software manager: Andreas Johansson Hardware Manager: Walter Lagerhäll
Date	2023-10-26

Table I
PROJECT INFORMATION

Name	**Title**
Andreas Johansson	Software manager
Walter Lagerhäll	Hardware manager
Kasra Ekrad	Software developer
Sebastian Leclerc	Software developer
Elon Pettersson	Software developer
Erik Rågberger	Software developer
Sharifeh Yaghoobi	Hardware developer

Table II
STAFF AND ROLES

II. Project Progress Summary

A snapshot of the project's progress during the first sprint can be seen in Table III. The starting date for the first sprint was 2 October 2023 and the duration was three weeks. The table shows how far each work package had progressed at the end of the first sprint. Each work package's status at this point and its tasks are discussed in detail in Section III.

In general, the project has been progressing, but there have been some hurdles and uncertainties. Main concerns have been regards the Nvidia Drive PX2 platform which could not be accessed until after one week into the first sprint. The high level system design is now almost complete regarding what devices to use where and how they are connected and communicate. The PX2 platform led to the data communication work package being off track since the whole systems has undergone revisions and redesigns. The data communication work package is one of the work packages with the heaviest work load in the project and most of the work will be pushed to sprint two.

Several of work packages that are not 100 % done, such as, emergency brake, UGV power management, tether wire specification are mainly waiting for delivery of components. These will be assembled and tested once they are delivered. Right now, they are considered on-track and no issues are foreseen with these work packages. Finishing them will be straight-forward.

Hardware work packages that are 50% done as UGV component placement, UAV power specification and UAV 3D printing are either waiting for hardware components or design files from collaboration partners in order to be fulfilled. After this the completion time for the tasks will be low.

Work package (sprint one)	Done (%)	Notes
Tether attachments	100 %	
Tether wire specification	95 %	
Data Communication	40 %	
Follow UGV "computer vision"	50 %	Work spans several sprints
UAV 3D Printing	50 %	Moved from sprint 2
UAV power specification	50%	
Control UGV with controller	100 %	
Emergency Brake	90 %	
UGV component placement	50%	Work spans several sprints
UGV power management	80 %	
Raspberry PI base environment	100 %	Extra work package

Table III
SPRINT ONE OVERVIEW (2-20 OCTOBER 2023)

III. Work Package Status

The following section discusses the status of every sprint one work package and their tasks in more detail. The section outlines progress, issues, next steps, and how the package is faring with regard to the project plan. Accountability and recovery plans are also addressed in this section. Table IV shows the ID for a specific work package, its current status, and the person accountable for its progression. Work packages can be referred to in the attached Project plan, see Appendix A, Section 5.

ID	Work package (sprint one)	Status	Accountability
1.1	Tether attachments	Completed	Andreas
1.2	Tether wire specification	In progress	Andreas
2.1.5	Data Communication	In progress	Sebastian
2.1.4	Follow UGV "computer vision"	In progress	Elon
2.2.2	UAV 3D Printing	In progress	Sharifeh
2.3.1	UAV power specification	In progress	Walter
4.3.1	Control UGV with controller	Completed	Kasra
4.3.3	Emergency Brake	In progress	Erik
4.4.2	UGV component placement	In progress	Walter
4.5.1	UGV power management	In progress	Sharifeh
x	Raspberry PI base environment	Completed	Sebastian

Table IV
WORK PACKAGE STATUS FOR SPRINT ONE (2-20 OCTOBER 2023)

A. Control UGV with controller

The *Control UGV with controller* work package is: on-track.

It was possible to utilize a wired Logitech controller connected to the ITX computer and make the Husky move. Therefore, we can use this method of controlling the Husky for the initial testing and demonstrations. Tests with sending similar ROS commands to move Husky also have been done successfully. The first purpose was to utilize PX2 to control the Husky. It was not possible to install the drivers for the same Logitech controller on the PX2 due to limitations and restrictions made by Nvidia on the PX2 device. PX2 has a list of compatible devices, sensors, and drivers which are only exploitable devices on PX2. In the end, ITX was chosen as a possible option to control the Husky with the controller. As this work package status is completed there will be no further operations on this task. However, another task was created after finishing this task with the title of "Controlling Husky with a wireless controller" which is put in the backlog. This task will be considered in the last sprint if all other higher-priority tasks are done.

B. Data Communication

The *Data Communication* work package is: off-track.

The Data communication work package consists of setting up a data communication framework, which later can be populated with the correct data. Tasks have been to set up ROS communication from the UAV's Raspberry Pi to the flight controller, establish a communication channel between the UGV and UAV, and implement a heartbeat monitor in case of broken connections.

This work package has been heavily revised during the sprint because of insufficient knowledge about the PX2 platform. The PX2 lacks Bluetooth support, which has necessitated a reevaluation of our communication strategy. This impacts not only the connection between UGV and UAV but also how we communicate with the flight controller, as the available interfaces vary depending on the chosen solution.

The status as of 20 October 2023 is that we have verified that Bluetooth communication works between the UAV Raspberry Pi and another Raspberry Pi which might be part of the UGV system. We will however test if we can utilize the existing ITX computer on the UGV for this Bluetooth link, thus eliminating the need for two Raspberry Pi's. It was not possible to use the PX2 for the Bluetooth link due to missing Unix kernel header files. We have also verified that the flight controller communicates with UAV Raspberry Pi via the ROS DDS protocol over an Ethernet link.

Since the work package is off-track some steps have been taken to get it on track again. We are allocating more resources to developing the software, ensuring that the package is prioritized and progressing. Due to our flexible planning, we are allowed to push work packages into later sprints without affecting the goal of the project. However, low-prioritized work packages classified as "nice-to-have", such as computer vision-functionalities on the UGV may be canceled and removed entirely in the later stages of the development phase.

Moving forward, we need to finalize the last communication links, e.g., how the PX2 will communicate with the rest of the system. We also need to test the end-to-end communication. For example, reading some sensory data on the UGV in the form of a "Hello-message", and passing the data through the ITX, via Bluetooth to the UAV who responds with some form of acknowledgment. Thus supporting bidirectional communication throughout the whole system.

C. Follow UGV "computer vision"

The *Follow UGV "computer vision"* work package is: On-track.

The camera part is already well on its way. Currently, an AI YOLO model exists that can locate two different targets where one is a red circle and one is a blue circle and they have different patterns in the center. It was meant to be a prototype but it works well enough to be used on the final product, and runs fast enough to be used on a Raspberry PI. Now, the next step is to additionally utilize the IMUs from the drone and the husky to get more precise movement and orientation data to be used for localization of the drone. This is being worked on at the moment and most of the effort is going into ROS2 programming. Exactly what information the flight controller expects to be able to be controlled from code is still uncertain, hence, research is currently being done to find a good way to establish this communication.

D. Emergency Brake

The *Emergency Brake* work package is: Off-track.

The work package is almost completed, components have been ordered and an investigation too what signals the husky sends while the e-stop is engaged has been done. The work package is currently waiting for the delivery of the emergency stop button that has been ordered. The work package will be completed once the emergency stop button has been mounted and connected to the husky.

E. UGV component placement

The *UGV component placement* work package is: On-track.

The UGV component placement includes placement of all components on the UGV platform. It is decided where all the specific components will be located on the platform. The placement of specific components was decided through meetings with representatives from the hardware team as well as the software team. Additionally the design of sensor attachments is also started. However, how to mount the sensors onto the UGV depends on the design of the Loading area and Landing pad and the task is therefore blocked until the design aspect of these elements are completed. Upcoming tasks for this work package include finalizing design of the sensor attachments and 3D print them and will be done once all necessary information is provided regarding the design of the loading area/landing-pad.

F. UGV power management

The *UGV power management* work package is: On-track.
The work package is not done yet, but all necessary equipment including battery to supply the PX2, charger for the battery, safety components, power button, and connector for the PX2 are investigated and placed on the order list. The work package will be complete as soon as the equipment are received, assembled and tested.

A task was defined at the beginning of the sprint to find a solution for power management of the husky and the PX2. The result indicated that the husky and PX2 works on different voltage levels. Additionally, the power socket embedded on the husky to connect a companion computer could not support the current and power needed for the PX2. Hence, it was decided to supply the PX2 by a separate battery and a task was defined to investigate an appropriate battery. Additionally, for safety of the equipment another task was defined to find components needed for isolating and securing the powering system of the PX2. The investigation was a study task about electrical circuits, battery, and elements needed in a circuit due to lack of knowledge about the PX2. At the end of investigation, some components including a Dc-Dc converter, low voltage disconnect, fuse, and diode were suggested. However, after a while, we got access to the technical data sheet of the PX2 where it was highly instructed that we have to use a fuse and diode for isolating and securing the PX2 in the case of powering by a vehicle battery. Hence, the recommended fuse and diode was ordered.

Another task in this work package was study about the power cable of the PX2 that uses a 10-pins ATX power connector to identify the purpose and voltage of each pin. Moreover, to make the powering system more practical, it was decided to use a power button(switch) to control the flow of the electrical power instead unplugging the 10 pin cable each time, therefore a task was defined in the work package to decide an appropriate switch for the application.

The status of the work package at 25 October is that we have received most of the ordered parts, and the work package will be subjected to assembling and testing phase soon.

G. UAV power specification

The *UAV power specification* work package is: Off-track.

The work package started with deciding and ordering all necessary components including battery, power cables and connectors in order to be able to supply the UAV from the ground. The work-package is currently blocked since important ordered components have not been received. Once the components have been received connectors will be soldered to the power cables. Since the cables up to the UAV will be 5 meters long a voltage drop test will also be conducted to verify that the thickness of the cables are enough for the power consumption of the UAV. Even though the work package is off track it will not affect upcoming tasks a lot. Once all components are received this task will be prioritized and performed as soon as possible.

H. Tether wire specification

The *Tether wire specification* work package is: on-track.

While the work package is not done, the USB cable, power cables, tether, and cable sock are decided and placed on the order sheet. Currently, we are waiting on the power cables and USB cable. After both are received, they will be tested and incorporated into the cable sock, ready for use.

I. Tether attachments

The *Tether attachments* work package is: on-track.

The attachments for the tether on both UGV and UAV are considered part of the tether solution. Currently, the UAV tether attachment design is outsourced to Bengt-Erik Gustafsson, the attachment is designed and will be included in the 3D-printing of the UAV. The tether attachment for the UGV is outsourced to Makers of Västerås. The attachment will be part of the landing pad which they are responsible for designing and building to our specification.

J. UAV 3D Printing

The *UAV 3D Printing* work package is: On-track.

The work package is in progress at 25 October, with some components of the UAV, such as the UAV body, arms, and legs having reached the final design and printing stages. However, other elements, like the hooks and mounting boards are still in the design phase. Once the design phase is complete, these components will be printed and subjected to assembling and testing.

The task was done for initial design during the sprint one, however, the design of the most of the elements like the body, arms, and legs had to be changed. Hence, the task moved to the sprint two for redesigning and printing.

K. Raspberry Pi base environment

The *Raspberry Pi base environment* work package is: on-track.

A standardized base environment for the Raspberry PI on UAV was created after much trial, error, and re-installing of the operating system occurred. An installation guide was created to quickly and correctly re-install the operating system with correct project dependencies when it breaks. A lot of effort went into trying to clone the SD cards but issues with sector sizes prolonged the duration of the task. However, the task is finished.

IV. Completed and Upcoming Activities

During the first sprint, several work packages were not able to be 100 percent completed as seen in Table IV. However, the project is progressing well and some milestones that have been achieved include:

- The first version of the drone design is finalized
- The UGV can be driven with a controller
- Communication has been established between all nodes
- The PX2 platform can be accessed
- Establishment of a workflow that produces good results regarding the computer vision algorithms for the drone

For sprint two, we are focusing on the work packages in Table V. The expected deadline for each work package is also listed in the table. Upcoming milestones would be to have the different devices communicate information properly as well as assemble and test-fly the drone. The design for the UGV landing pad and loading area will hopefully be finished and their respective building phase started.

ID	Work package	Notes	Deadline
1.2	Wire specification	Cont. from sprint one	Sprint two
2.1.4	"Follow Husky"	Cont. from sprint one	Sprint three
2.1.5	Data communication	Cont. from sprint one	Sprint three
2.2.1	UAV Electronics	Starts in sprint two	Sprint two
2.3.1	UAV Power spec.	Cont. from sprint one	Sprint two
2.2.2	UAV 3D printing	Cont. from sprint one	Sprint two
2.3.2	UAV Assembly	Starts in sprint two	Sprint two
2.3.3	Battery level indicator	Starts in sprint two	Sprint two
4.3.3	UGV Emergency break	Cont. from sprint one	Sprint two
4.4.1	UGV Landing pad	Starts in sprint two	Sprint three
4.4.2	UGV Component place.	Cont. from sprint one	Sprint two
4.4.3	UGV Loading area	Starts in sprint two	Sprint three
4.5.1	UGV Power mgmt.	Cont. from sprint one	Sprint two
4.5.2	UGV Sensor equipm.	Starts in sprint two	Sprint three

Table V
UPCOMING WORK PACKAGES FOR SPRINT TWO

V. ISSUES, BLOCKERS, AND RISKS STATUS

An issue that have occurred is that the Nvidia PX2 was harder to work with than expected. Documentation is limited, vague and was not accessible in the beginning of the sprint due to problems with licenses. Furthermore, some of the ports of the computer are currently blocked to receive data from which will affect the usage of several sensors. To solve this problem the Mini ITX computer will be added to the system to perform some of the tasks not requiring heavy processing power. Also an investigation will be made during second sprint to identify sensors that can easily interface with the PX2 and consultation with stakeholders will determine the feasibility of adding these sensors to the platform.

Another significant issue and blocker is that some important hardware components have not been received by the end date of the sprint. This clearly delays the delivery of some hardware work packages and also a few software work packages in the future. While the majority of the hardware components were received during the week following the first sprint, the crucial ones that remain outstanding are expected to arrive shortly. Although the execution of some work packages have been temporarily blocked due to the pending component delivery, we can plan to prioritize their implementation in the second sprint once the components are available.

A potential hardware related risk involves the possibility of missing to order an essential hardware component or ordering the wrong component. If this happens the responsible persons for ordering components will be contacted immediately to ensure that the new component will be ordered and delivered as soon as possible.

VI. INDIVIDUAL CONTRIBUTIONS

The following section summarizes what each project member has been working on during the duration of sprint one.

A. Kasra Ekrad

By the time Sprint One started, most of Kasra's responsibilities were in relation to the control system as this is his main role in the project. Thus started to learn the ROS2 framework at start of the sprint One.

After gaining a basic understanding of how the ROS system works, started to work on the "Check compatibility of Ubuntu/ROS distributions on all nodes." task by installing and using ROS commands on different distributions of ROS and different Ubuntu distributions based on project's software and hardware requirements. Then, started to investigate using Docker by installing and running several nodes on different Dockerized systems to mitigate ROS distribution problems on the "Ros 2 Docker" task. The idea of having Docker containers on different systems was discarded after a meeting with supervisors.

"Installing and evaluating Ros kinetic version on ubuntu 16" task was the further topic that had to be investigated after those changes. Worked on installing ROS and UGV packages on Ubuntu 16.04 which is the PX2 operating system. Another task that Kasra was involved in was "Control with Controller" which first was the PX2 that meant to move the UGV by applying previous tasks phases on PX2. But it was not possible. In the end, checking if the ITX computer located on UGV is able to move the UGV with the connected controller was done. Next Kasra's task was "Investigate emergency break topic". The corresponding command and its topic were found and checked if they are functional.

Furthermore, made a contribution to the "ROS communication via Ethernet between MC and FC" task helping Andreas follow the flight controller's vendor instructions.

After all, started to access PX2 device functionality on the "Prepare PX2 for development" task. By applying for the Nvidia development license and gaining access to the PX2 device documentation and operating system. During this task, the probability of using available Lidar sensors was assessed. Moreover, contributed to the "Create base environment for pi" task and also created several clones of the Raspberry Pi based on the instruction document which Andreas, Kasra, and Sebastian provided during this task.

In the end, participated in the "Send ROS command from RPi to FC" task where ROS nodes on Rasberry Pi were developed and some initial code for the nodes was provided to be able to communicate and handed it to Elon to continue completing the node's functionality on the UAV. "Connect PX2 to Husky Physically", "Install Controller on PX2", and "Running Ros2 Husky packages on PX2" were the final tasks in spring one that were contributed.

B. Andreas Johansson

At the beginning of the sprint, a lot of tasks were blocked because we had no access to the PX2 platform, and had not received any Raspberry Pi to develop on. Therefore, initial work was done on object detection by creating a test data set and exploring options such as Roboflow, to streamline the process.

In collaboration with Sebastian, mainly worked on designing and deploying the communication scheme throughout the system. Consideration was taken from flight controller on the UAV to PX2 on the UGV. Data from the 360 camera was also accounted for in this work. The communication behavior and data that will be sent between nodes will be added later, but the framework is created and every part is confirmed working as intended for now. Was also solely responsible for tether system related tasks such as specifying, ordering, assembling the cable package, and ensuring the tether attachment would exist on the UAV.

C. Walter Lagerhäll

The main focus during the first sprint have been to find and order hardware components suitable for the project. Critical hardware components have been identified and prioritized in the ordering to enable the execution of important work packages.

Another responsibility has been to communicate with the cooperation partners Bengt-Erik Gustafsson and Makers of Västerås. Bengt-Erik Gustafsson has been entrusted with the task to design of the drone following our specifications. Meanwhile, Makers of Västerås will be responsible for the design and manufacturing of the loading area as well as the landing pad according to our guidelines. Although the part with Makers of Västerås will be done during the next sprint it is of great importance to start the communication with them to ensure that the task will be delivered during.

D. Sebastian Leclerc

One of the main responsibilities has been to figure out how the various systems should connect and communicate with each other. With the help of Andreas, we successfully set up, so far one-way, Bluetooth communication between the ground and air systems.

Another responsibility was within setting up the Raspberry Pi base. After trying several different times to clone an existing SD card with a working OS, onto another SD, a time-consuming process, it was instead finally decided to set up a guide of all the necessary dependencies for the UAV's Raspberry Pi.

Have also aided teammates in many different tasks, such as helping out in different Linux environments. Have also read and documented the major parts of how to build a UAV from scratch, in order to help decide the various components that the UAV should consist of. This consisted of investigating what flight controller to get, how it must communicate with a peripheral computer (Raspberry Pi), and what other peripherals we must get (Radio receiver, transmitter, GPS, power distribution, etc).

E. Erik Rågberger

The main area of responsibility has been the emergency stop button for the husky. Investigated what dimensions etc would fit the husky as well as investigating what is happening in the husky when the emergency stop is engaged.

Otherwise Erik has been responsible for contact with Nvidia about licenses and other than that he has been looking into calibrating a Vectornav 200 and applying a ROS package to it so that it can communicate using ROS.

Erik has also been aiding other team members in tasks such as helping the hardware team with investigations, creating data sets for the follow husky and communicating with Makers of västerås about the landing pads.

F. Elon Pettersson

The main area of responsibility has been in the Follow UGV "computer vision" task. Worked on creating landing pad targets, and printing them. Together with Erik, the data-set was created, consisting of the targets in different locations, angles, and distances from the camera. The YOLO machine-learning model was created on Roboflow, and a code was written to find the angle between the targets. Making this program communicate with other nodes is the next part of this process.

The focus is currently on ROS 2 communication. The communication (publishing/subscribing) between nodes is currently being worked on as well as utilizing IMU data to add additional certainty to the localization of the drone relative to the husky. The node structure is already planned and research is also currently being done on how to tell the flight-controller what to do.

There have been a lot of issues regarding computers so much time has been spent on installing and reinstalling dependencies regarding ROS and Python as well as Ubuntu. Several virtual machines has been installed and used in this process. Creating packages and files in ROS has also required much bug-fixing. The build generation files used (CMake and XML) has created huge problems for various reasons, requiring some work to be redone several times, resulting in time loss.

Elon is also in charge of a system specification document that specifies all components and cables that makes up the system, and their statuses such as, decided or not, or alternative components.

G. Sharifeh Yaghoobi

Throughout the sprint, One of the responsibilities was searching and reviewing related works of the project to identify the approaches and necessary sensors for localization, navigation, and object detection by the UGV as well as investigation of the existing sensors both from the last semesters and on-built sensors. Moreover, Sharifeh investigated designing of a ground station for the tethering system to make the retraction and expand of the rope or power cable automatic by a winch system. After investigation, the design of the system identified as a low priority work package, therefore, it left as a future work because the task was time consuming and no requirement for the first iteration of the project.

Sharife was also responsible to investigate the power management of the UGV including battery power supply of the PX2, power button, safety components and other. The investigation phase was completed during the time line, however, the assembly phase will be blocked until we receive the components.

Another responsibility was 3D printing of the UAV elements and sensor placement on the UGV. The task was done for the initial design, but, the design of the some elements had to be changed. The task will be finished when designing phase of the elements are finalized and printed.

References

Appendix

On the following page, the project plan is attached as an appendix.

Project plan

Project Name	Stakeholders/Sponsors	Project Manager
Safe Trustworthy Autonomous Delivery System	Mälardalen University Volvo CE Alstom	Software manager: Andreas Johansson Hardware Manager: Walter Lagerhäll

Table 1: Project information

Name	Title
Andreas Johansson	Software manager
Walter Lagerhäll	Hardware manager
Kasra Ekrad	Software developer
Sebastian Leclerc	Software developer
Elon Petterson	Software developer
Erik Rågberger	Software developer
Sharifeh Yaghoobi	Hardware developer

Table 2: Staff and Roles

1 Project Information

The Safe Trustworthy Autonomous Delivery System (STAD) project is a multi-year project at Mälardalen University. In this project, an Unmanned Aerial Vehicle (UAV) is tethered to an Unmanned Ground Vehicle (UGV) robot to perform autonomous deliveries in the city environment, namely, Finnslätten in Västerås. Finnslätten is an industrial area with many industries and factories located within it. Since this area is evolving into a technological hub, access to the public roads will be restricted, therefore, it requires an alternative solution for shipments, such as UGV robots instead of ordinary vehicles. However, safety and trustworthiness are two significant concerns regarding the UGV solution due to the presence of diverse obstacles such as humans, trees, buildings, and more on local passages in the area. With regards to the fact that obstacles can vary in type, shape, size, and location the robot needs to have a wide and clear view to detect any type of obstacles in an acceptable timeframe and avoid them in a safe manner as well.

Research has been conducted on obstacle detection methods of UGVs [1]. The results of other studies show that UGV has limitations in supporting a wide range of sensors, operation in confined and unstructured environments, and other due to lack of field of view[1], [2]. According [2] to address the limitations of an UGV, an UAV can be used to provide a wide field of view. Related researches have indicated the utility of having an UAV assisting the UGV as a visual assistance for infrastructure inspections and surveillance tasks [2],[3] and [4]. However, UAVs have limited onboard power capacity that make the platforms inappropriate for long missions. According [5], removing the battery from the UAV and directly supplying power from the UGV using a tether, make the platform appropriate for long missions as well. As the result, a platform consisting of an UAV tethered to an UGV will be developed to do the shipment task in the urban environment.

This is an interesting area, providing value to automotive industries in similar applications. In construction sites and similar areas, as mentioned, a vehicle must be equipped with various sensors to get clear sight of the surrounding environment, nonetheless, a challenge arises as these sensors can become obstructed by the machine's own body parts. Excavators for instance will cover different sensors depending on where its arm is. This can be a hazard, and in these cases, using a tethered UAV could be a valuable asset for providing a surveillance overview and infrastructure inspection as it solves the problem of the view being obstructed [3].

Volvo CE is a company that is exploring autonomous solutions and they have provided an OAK-D [1] depth camera for this project, which will be used by the UAV to locate the UGV. Another company that is showing great interest is Alstom which is a company working with train transport systems. Alstom has sponsored the project with an Nvidia Drive PX2 [2] which is a computer platform specially made for image processing used in autonomous vehicles. The PX2 will be a valuable asset as a direct result of this project's heavy reliance on image processing.

In this iteration of the project, the sensors will be mounted onto the UGV, and the UAV will be built as well as tethered to the UGV. Additionally, the UAV will be given the semi-autonomous ability to follow the UGV by itself, utilizing a camera lock on a pattern located at the UGV, and by deploying vision algorithms. At the end of this iteration, the goal is to build a fully functional hardware complete platform in which software can now be deployed to develop algorithms and using the mounted sensors to make the system fully autonomous.

2 Purpose and Goal

Finnslätten, Västerås is evolving into a technological hub where access to public roads is prohibited or restricted. The purpose of this project is to provide autonomous, safe, robust, and reliable transportation of products in urban environments that are experiencing these limitations.

The goal of this project is to manually drive an Unmanned Ground Vehicle (UGV) 250 m on a flat surface. A tethered unmanned Aerial Vehicle (UAV) will be attached to the UGV and autonomously follow it. The development of the project will be limited between 2023-10-02 to 2023-12-14.

3 Limitations

The limitations of the project are stated below. It is important to mention that this project is a part of a larger project. Some of the limitations in this project will be remedied in the later stages of the larger project.

1. The project will not produce a final product.
2. It is not guaranteed that the product is free from bugs and will work under all circumstances.
3. It is not guaranteed that the product is entirely safe.
4. It is not guaranteed that the product can be tested as desired due to potential bad weather conditions.
5. The operation time will be limited to the capacity of the batteries.
6. The drone system will not be able to automatically detect obstacles and avoid them.
7. The project can not guarantee emergency landing functionality, in the event of power loss to the drone.
8. It is not guaranteed that the system will be fully autonomous.
9. The project will not deliver a functioning winch system.
10. The project will not produce a completely trustworthy product.
11. The RICOH Theta X camera will only be equipped on the drone and transfer images to the GPU of the Husky, it is not guaranteed that the images will be processed.
12. The system will be implemented with a flat surface in mind, inclined paths will not be taken into consideration.
13. The drone's navigating and positioning vision might be obstructed by objects and the tether. As a result, the positioning algorithm might decide unreasonable movements while the drone is obstructed.

4 Requirements

The STAD project is undertaken as a proof of concept (POC) to explore the feasibility of an autonomous delivery system. It is important to note that this is the first iteration of the STAD project and the requirements are intentionally kept flexible. The flexibility allows us to be adaptable and reach the goal within the allotted time frame.

[1] https://shop.luxonis.com/products/oak-d
[2] https://docs.nvidia.com/drive/active/5.0.10.3L/nvvib$_{d}$*ocs/index.html*

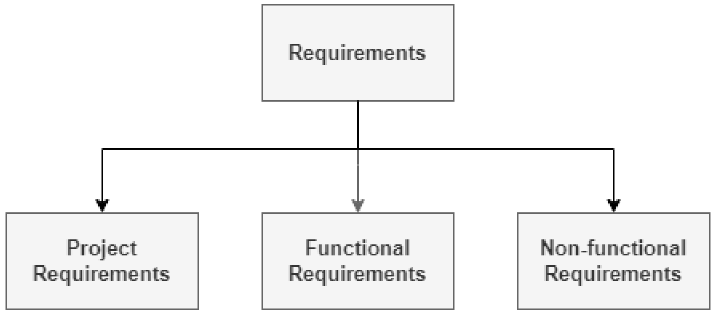

Figure 1: Projects Requirements

4.1 Project Requirements

1. The project shall be completed before the development phase ends on 2024-12-14.
2. The project shall use the following Git repository[3] for software development and version control.
3. The project shall also provide assignment documents such as project plans and reports to the Git repository.
4. The project shall be managed and planned according to already existing project management principles, specifically this project will follow an Agile approach with Scrum methodology.
5. Codes, depending on language, shall be formatted according to one style guide per language.
6. The system shall consist of a Husky UGV and a UAV in the form of a quadrocopter drone.
7. The UGV shall be equipped with a lidar, and at minimumof two cameras for future research purposes.
8. The system shall have a power connection between the UGV and UAV.
9. The UAV shall be equipped with a Ricoh Theta X 360° camera for future research purposes.

4.2 Functional Requirements

1. The UAV shall provide a live stream from the Ricoh Theta X 360° camera to the UGV.
2. The UAV shall also be equipped with a regular camera to locate and track the UGV.
3. The UGV and UAV shall support two-way communication.
4. The UAV shall have an emergency stop and landing functionality.
5. The UAV shall be able to be controlled manually using a radio.
6. The UGV shall have a landing pad for the UAV.
7. The UGV shall have emergency stop functionality.
8. The UGV shall be able to access data from the extra-equipped sensors.
9. The UGV shall be able to be directly controlled by a user via a game-pad.

4.3 Non-functional Requirements

1. The UAV shall autonomously follow the UGV, staying centered above it within a two-meter radius.
2. The UAV shall be tethered to the UGV using a wire with a maximum length of five meters. Other cables shall have a maximum length of 25 cm less than the tether.
3. The system shall be able to make one 250 m trip on a flat surface.
4. The UGV and the UAV shall manage 10 minutes of continuous operation.
5. The UGV shall have a loading area capable of carrying a maximum of 40kg.
6. The system shall have an emergency stop so that both vehicles stop within 500 ms (UGV stops moving, UAV hovers in place).
7. The UAV motors and power system shall allow a minimum lifting capacity of the UAV itself and its connected cables.

[3] https://github.com/MDU-C2/Shuttle

5 Work breakdown structure

The following work breakdown structure is formatted accordingly in order of size: Root > Major deliverables > Sub-deliverables > Work packages > Tasks. During sprint planning and sprints, work packages may be broken down even further into tasks and sub-tasks. In doing so, extra granularity will be provided and make it easier to keep track of individual tasks progression.

Root- Safe Trustworthy Autonomous Delivery System

1. Major deliverable - Tether System
    - 1.1. Work package - Attachments
    - 1.2. Work package - Wire specification
2. Major deliverable - Drone/UAV
    - 2.1. Sub-deliverable - Control System
        - 2.1.1. Work package - Flight control
        - 2.1.2. Work package - Lift by ground control
        - 2.1.3. Work package - Emergency landing by ground control
        - 2.1.4. Work package - "Follow Husky" computer vision
        - 2.1.5. Work package - Communication
            - 2.1.5.1. Task - ROS-communication between MC and FC
            - 2.1.5.2. Task - Bluetooth Communication Protocol
            - 2.1.5.3. Task - Hello messages
    - 2.2. Sub-deliverable - CAD Design
        - 2.2.1. Work package - Electronics cabling placement
        - 2.2.2. Work package - 3D printing
    - 2.3. Sub-deliverable - Hardware/electronics
        - 2.3.1. Work package - Power specification
        - 2.3.2. Work package - Assembly
        - 2.3.3. Work package - Battery level indicator
3. Major deliverable - Documentation
    - 3.1. Work package - Datasheet compilations
    - 3.2. Work package - Hardware setup guide
    - 3.3. Work package - Logic/SW flowcharts
4. Major deliverable - UGV Husky
    - 4.1. Sub-deliverable - Navigation
        - 4.1.1. Work package - Localization
        - 4.1.2. Work package - Path Planning/Path following
        - 4.1.3. Work package - Create Map
        - 4.1.4. Work package - Husky Orientation
        - 4.1.5. Work package - Gather odometry data
    - 4.2. Sub-deliverable - Vision
        - 4.2.1. Work package - Camera/Lidar connections
        - 4.2.2. Work package - Object detection corner cameras
        - 4.2.3. Work package - Object detection lidar
        - 4.2.4. Work package - Road tracking corner cameras
    - 4.3. Sub-deliverable - Control system
        - 4.3.1. Work package - Motion control
        - 4.3.2. Work package - Anti-collision system
        - 4.3.3. Work package - Emergency break
    - 4.4. Sub-deliverable - CAD Design
        - 4.4.1. Work package - Landing pad

4.4.2. Work package - Component placement
4.4.3. Work package - Loading area
4.4.4. Work package - 3D printing
4.5. Sub-deliverable - Electronics
4.5.1. Work package - Power management
4.5.2. Work package - Equip Husky with sensors

6 Project Schedule

The project is roughly planned into four sprints, the sprints are three weeks long except for the final sprint which is two weeks long. Before each sprint, a sprint planning meeting will be held to finalize the current sprint plan. Work packages will be broken down into specific tasks, feasibility of the current sprint plan will also be considered. After each sprint, a sprint review meeting will be held where completed tasks will be discussed as well handling of incomplete tasks.

Every task has been time estimated with three point estimation[4] by the whole team. The estimations subsequently have been compiled into work hours, as shown in Figure 2 to Figure 6. Each sprint has the capacity of circa 15 weeks of work hours. The most critical tasks are handled in sprints one and two. Sprints three and four will handle tasks of low priority and tasks that are not dependent on requirements. It will also be possible to push delayed tasks to later sprints since these sprints are under-allocated to critical tasks. After all development sprints are done, there will be a handover of the projects.

Figure 2: Sprint 1

[4] https://en.wikipedia.org/wiki/Three-point_estimation

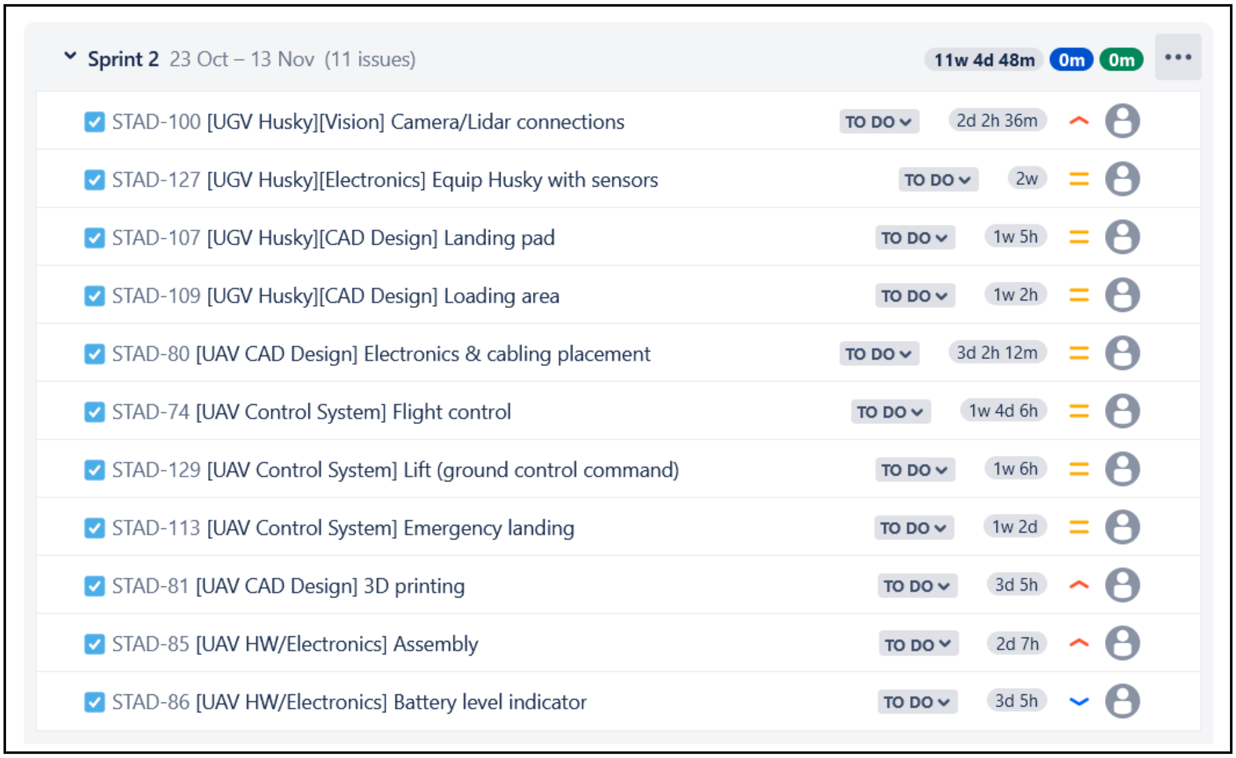

Figure 3: Sprint 2

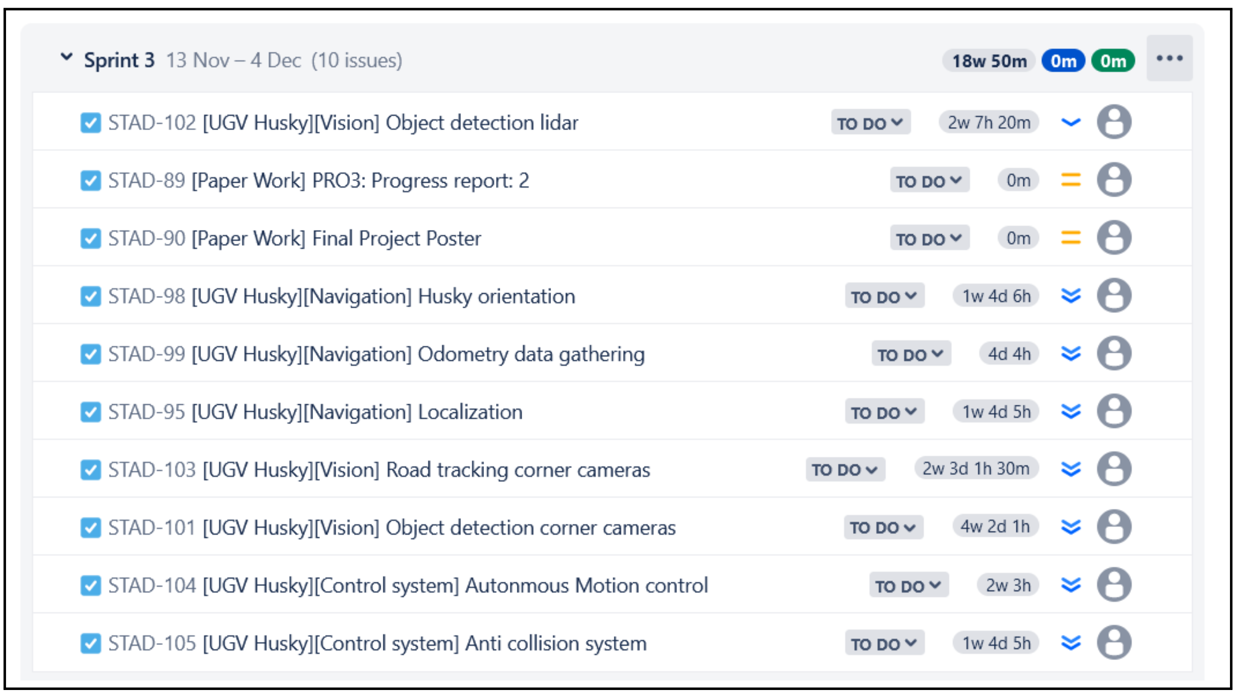

Figure 4: Sprint 3

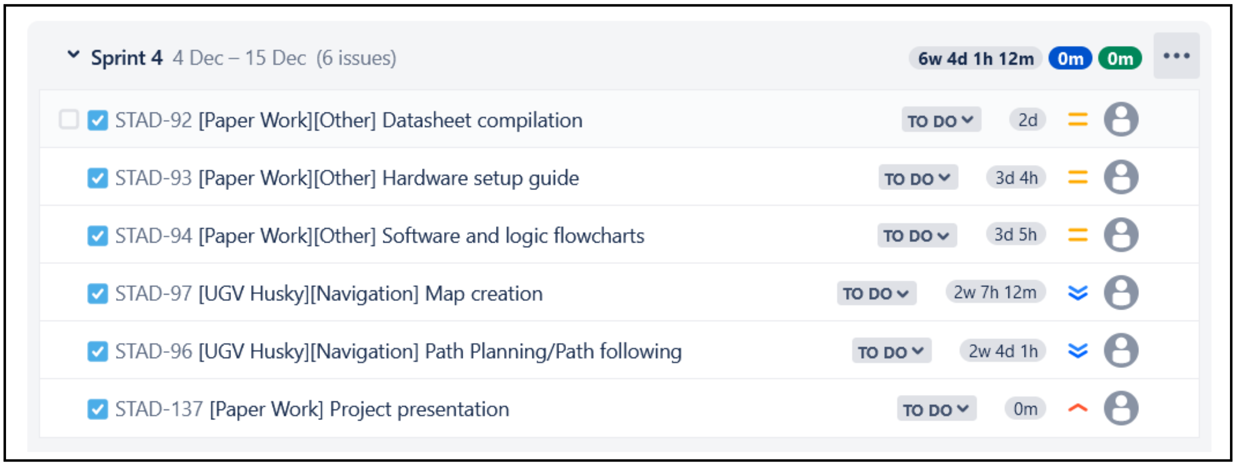

Figure 5: Sprint 4

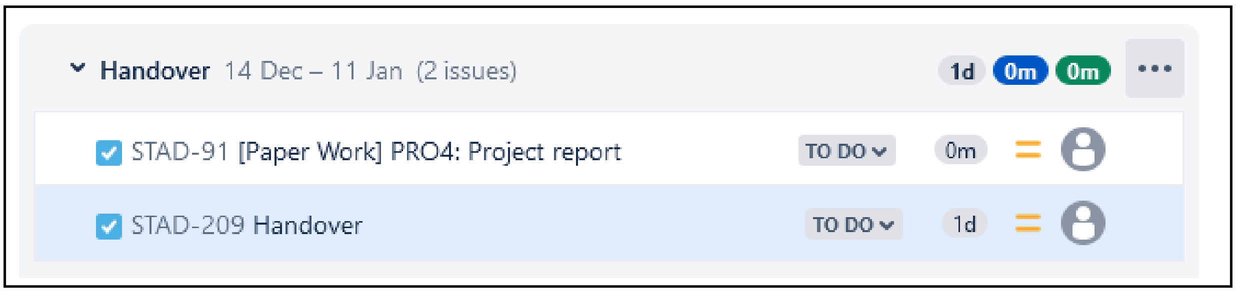

Figure 6: Handover

7 Development Plan

This Development Plan provides a comprehensive road-map for the project, detailing our technical approach, the development process, the methodologies we employ, and each team member's distinct roles and responsibilities. The plan is designed to facilitate seamless integration for new members joining at any project life-cycle stage. By becoming familiar with the plan, crucial insights into this project's work processes and project-related information, enabling anyone to contribute effectively and align with the team's objectives.

7.1 Development Methodology

The agile methodology adapted for development is used in this project. The main reason for employing Agile methodology is that Agile is highly adaptable to changing requirements and priorities since this project has an innovative and unpredictable in environment. The methodology provides us with smooth communication between participants as well as flexible task changes among different components and developers. Agile gives us the ability to form cross-functional team roles as testers, software and hardware developers as well as software and hardware managers. All tasks and main stories are broken down into discrete pieces of functional tasks including expectations and their priorities. Afterward, four sprints were planned as described in Section 6.

7.2 Team Structure and Roles

The project's team structure is specifically crafted to leverage the distinctive skills and capabilities of each team member, fostering effective teamwork. The roles listed here might have additional responsibilities outlined in the corresponding plans. In addition, all the participants collaborate in documentation tasks.

- **Hardware Team Leader:** Manages the hardware development process, coordinates with the software team, and ensures timely delivery while maintaining quality standards.
- **Software Team Leader:** Oversee the software development process, collaborate with the hardware team, and ensure project deadlines are met with high-quality deliverables. Plans the project's sprints, and tasks, and assigns them to the responsible person.
- **Hardware Developer:** Designs, develops, tests, and troubleshoots hardware components and collaborates with the software team.
- **Software Developer:** Develops, tests, and debugs software modules and coordinates with the hardware team.

7.3 Tools, Technologies, and Systems

The staffing plan and assigned roles and responsibilities to each member of the project's team are described in Table 3.

Name	Title	Main responsibilities
Andreas Johansson	Software manager	Scrum master, data communication and computer vision
Walter Lagerhäll	Hardware manager	Scrum master, electronics and modeling
Kasra Ekrad	Software developer	Control systems
Sebastian Leclerc	Software developer	Data communication and control systems
Elon Petterson	Software developer	Computer vision algorithms
Erik Rågberger	Software developer	Computer vision algorithms
Sharifeh Yaghoobi	Hardware developer	Electronics and modeling

Table 3: Team roles and responsibilities

Roles were assigned to each individual based on their skills, experience, and interest in the project. Regular meetings will be held to ensure that everyone is on track and to address any concerns that may arise.

7.3 Tools, Technologies, and Systems

The selection of tools was driven by the project's specific requirements and constraints. During the tool selection phase, careful consideration was given to factors such as Husky and the UAV's specifications. Our primary focus was to identify the most suitable tools by evaluating the outcomes of the various options that were investigated. The employed tools, technologies, and systems in this project are as follows:

7.3.1 Hardware

- **Nvidia Drive PX2:** The NVIDIA Drive PX2 device is provided along with the Husky as a powerful embedded computing platform. It is designed for autonomous driving applications that require quick decision-making in different situations. It includes a range of compact and energy-efficient Graphics Processing Unit accelerated computing devices. The platform also leverages NVIDIA's deep learning technology for tasks such as object recognition, path planning, and decision-making. The platform supports sensor fusion, enabling the vehicle a comprehensive and detailed view of its environment. The Drive PX2 computing unit model is called AutoChauffeur, it is equipped with two Tegra Parker system-on-a-chip (SoC) modules. The platform includes 8 GB of Low-Power Double Data Rate (LPDDR) memory. The AutoChauffeur also includes 256 GB SSDs. Ubuntu 16 is installed as the operating system on this platform. Included ports are multiple Gigabit Ethernet ports, USB 3.0 and 2.0, CAN, UART, GPIO, specialized camera interfaces such as MIPI CSI-2, and other general-purpose ports. This device will be the principal computational node on the UGV. It will communicate with the flight controller or peripheral computer on the UAV to send commands. This device will receive a video stream from the UAV's 360° camera, the camera located on the UGV, data from the Lidar sensor, and all the other sensors connected to the UGV.

- **RPLidar A2 360° laser range scanner:** The RPLidar A2 utilizes a laser-based time-of-flight scanning technology to measure distances to objects in its surroundings. It performs full 360° scans with a range of up to 6 m. It is designed for various applications, including robotics, autonomous navigation, mapping, and obstacle detection. It provides data in the form of a 2D point cloud, which represents the distances and angles of detected objects in its field of view. It is compatible with various platforms and programming languages, including Robot Operating System (ROS), Python, C++, etc. The developer, SLAMTEC, provides a software development kit and documentation for developers to work with the RPLidar A2.

- **Husky A200:** The Husky is a medium-sized robotic development platform developed by Clearpath. Its large payload capacity and power systems are customized to meet research needs. Various peripherals such as cameras, Lidar, GPS, IMUs, and manipulators can be added to the UGV. The Husky is fully supported by ROS with community-driven open-source code. It comes with four motors, each equipped with a Quadrature encoder with 78,000 pulses/m sample rate, a 24 V 20 Ah Battery with 192W total power available.

- **Landing pad and Loading area:** A simple design for both the landing and loading will be constructed. The landing pad needs to be stable enough for the UAV to land without interfering with any electronics on the UGV. The loading area needs to be constructed for prototyping the actual goods being delivered.

- **3D printer:** The 3D printer available in the project is the Original Prusa i3 MK3 which has been tested with PLA filament. The printer will mainly be used to print the various parts of the UAV.

- **Luxonis OAK-D camera:** The OAK-D camera has its own processing power for AI operations using custom ML models. It is capable of calculating stereo depth and object tracking with its sensors. The picture quality depends on the framerate, which tops at 1080P using 60 FPS. Due to these capabilities, the OAK-D camera was chosen and will be mounted on the UAV with a downward-facing perspective to assist in tracking the UGV. It will be also used in the landing mode of the UAV to detect the correct position of the UGV's landing pad.

- **UAV:** A customized UAV will be created in this project which is tethered to the UGV. The power supply for the UAV will be located on the UGV and connected via a power cord. The UAV includes a Pixhawk 6x flight controller, a Theta X 360° camera, a PM03D power distribution board, four T-Motor 2820-7 motors, four Heli 20A speed controllers, and a 3D-printed UAV frame. The tether system consists of a pair of power cables, a USB cable, a sturdy rope or wire to protect the connections, and optionally cable sleeves to keep all cables together.

- **Flight Controller:** The Holybro Pixhawk 6X is used as the UAV's flight controller. It is equipped with advanced features for controlling autonomous vehicles, especially drones, suitable for both academic and commercial applications. It is equipped with an STMicroelectronics-based STM32H753 processor with a powerful Arm Cortex-M7 core running at up to 480 MHz, 2 MB flash memory, and 1 MB RAM. It also includes multiple onboard IMUs and barometers. The flight controller has modular input and outputs such as PWM, different radio control channels, RSSI, GPIO, GPS, I2C, Ethernet, SPI, and CAN Bus. In this project, the flight controller receives commands from a peripheral computer located on the UAV to determine its location and movement based on vision algorithm outputs.

- **360° Ricoh Theta X camera:** The camera has a 360° field-of-view for both images and video. It can capture images up to approximately 60 MP, 5.7 K resolution, and 30 fps video. It has an internal GPS module that can be acquired to embed into 360° photos. It has a USB-C interface and wireless connection to transfer the data. In this project, the camera will be placed on the tethered UAV and transfer its video stream via USB connection to the PX2 on the UGV.

- **Raspberry Pi 4B:** The Raspberry Pi will be deployed on the UAV to serve as a dedicated image processing node. This Raspberry Pi-based solution enables real-time image analysis and decision-making during flight operations. By offloading image processing to the Raspberry Pi, the computational load on the flight controller is reduced, aiding in a more stable flight performance. The Raspberry Pi 4B is also equipped with Bluetooth, allowing it to receive emergency stop signals from a remote control or ground station. To centralize the UAV's position relative to a UGV, the Raspberry Pi will be integrated into the ROS framework. It will send ROS commands to the UAV's flight controller, instructing it to adjust its position and flight path as needed to align with the UGV's location and objectives.

7.3.2 Software

- **File manager tool:** In this project development workflow, Microsoft OneDrive has chosen to leverage as the primary file management solution. OneDrive offers a robust and secure cloud-based platform that aligns perfectly with our project's needs.

- **Version Control system:** In this project, Git is employed as a version control system on the GitHub platform. Git is one of the most popular and widely used distributed version control systems. It is known for its speed, flexibility, and robust branching and merging capabilities in both small and large projects. It is used to manage changes to this project's files and maintain different versions of the project's software.

- **3D Design and printing Tool:** SolidWorks is a popular 3D computer-aided design (CAD) software that is widely used for 3D design and engineering tasks. Solidworks will be used to design and print the complete frame for the UAV.

- **Project Management Tool:** Within the group, Atlassian Jira will play a dual role, serving as both a task management tool and as an Agile Project Management solution specifically tailored for Scrum team management and project timeline tracking. It can be downloaded from it website.

- **Communication Tool:** For communication both within the project group and with stakeholders, Microsoft Teams and email will serve as the designated tools for sharing and receiving information, which can be downloaded here.

- **Operating Systems:** The choice of an operating system for participants' computers depends on their specific requirements, with options including Windows 10, Windows 11, or some Linux distribution.

7.3.3 Software Development Tools

- **Programming languages:** Various programming languages are used to develop hardware and software modules depending on their underlying requirements, documentation, and recommendations. C/C++ is used for programming hardware devices and can be downloaded from its source. Python and C++ are used to develop vision and machine learning algorithms as well as NVIDIA vision algorithms and interfaces.

- **Integrated Development Environment (IDEs):** Visual Studio Code (VS Code) has been chosen as the primary IDE for developing the software modules that will drive our project. VS Code is a versatile and powerful code editor that offers a wide range of features and extensions, making it an ideal choice for our development needs.

7.4 Testing and Quality Assurance

Quality assurance and testing are seamlessly integrated into our development tasks, forming an iterative and repetitive process. This approach will help mitigate software bugs early on and provide quality throughout the project's life cycle. A detailed testing plan is described in Section 11. Requirements gathering, design, and development discussions to ensure quality considerations are addressed at every stage. Test cases and test plans are developed in the planning stage along with the formation of the tasks. All the software components will be documented throughout the development process with the following rules:

- Try to comment functions as block comments, containing parameters and the general functionality of the method.
- Write single inline comments for single-line descriptions only if for complex lines.
- Comments should not duplicate the code.
- Do not excuse unclear code. Variable names should be clear themselves.
- Provide links to the original source of the copied code.
- Include links to external references where they will be most helpful.
- Add comments when fixing bugs.
- Use comments to mark incomplete implementations with TODO: comment.

7.4.1 Coding Standards and Guidelines

PEP 8 will be used as a coding standard for Python code to ensure high code quality and readability. The Autopep8 extension in VS Code will be used to automatically format Python code to conform to PEP 8 standards. For C/C++ coding, the CLang and CppCheck static analyzers will be used with the help of the C/C++ Advanced Lint extension in VS code.

7.5 Integration, Deployment, and Versioning

- **Integration:** For integrating software sections together, the first important phase is Unit Testing where individual components or modules will be tested in isolation to ensure that they perform as expected. Then components are gradually combined, and their interactions are tested to identify and resolve any issues related to data flow, communication, and compatibility between modules. This process will be repeated until the whole system is functional and can fulfill its mission.
- **Deployment:** Deployment will be done after an integration test within all listed hardware and software in Section 7.3. All the final software will be built and versioned based on the below rules. Afterward, final software versions will be installed on target machines. Required configuration or calibration should be finalized on each device such as UGV's PX2 board, the UAV's flight controller, and peripheral processors. Complete information about the calibration and configuration of software and hardware is available in our file management system.
- **Versioning:** The version of each release has two parts 1- major and 2- minor, e.g., 1.2 (1 for major and 2 for minor).
  - Major versions represent adding new features and/or making considerable changes in already existing components.
  - Minor versions represent bug-fixing versions and minor refactoring which does not affect the functionality of software components.

8 Communication plan

The communication plan encapsulates what types of communication are needed to succeed with this project. See Table 4 for the specifics of how this is done.

- **Who (Target Audience):** This determines who needs to receive specific information.
- **Why (Purpose):** This refers to the reason behind the communication.
- **What (Type of Information):** This refers to the specific data or information to be shared.
- **When (Timing):** This indicates when and how often the communication should occur.

- **How (Communication Channels):** This determines how the information will be delivered.
- **Responsible:** This identifies who is responsible for ensuring the communication happens.

Who	Why	What	When	How	Responsible
HW Team Leader	To Stay Informed On Project Status	Project Updates (Hardware)	Weekly	Email, Jira, Teams, Physical Meeting	Project Team
SW Team Leader	To Stay Informed On Project Status	Project Updates (Software)	Weekly	Email, Jira, Teams, Physical Meeting	Project Team
HW Team Leader	To Prepare For Meetings	Meeting Agendas	Before Each Meeting	Teams	Meeting Arranger
SW Team Leader	To Prepare For Meetings	Meeting Agendas	Before Each Meeting	Teams	Meeting Arranger
HW Developers	To Understand Their Tasks	Task Assignments	As Needed	Teams, Jira	HW And SW Team Leaders
SW Developers	To Understand Their Tasks	Task Assignments	As Needed	Teams, Jira	HW And SW Team Leaders
HW Developers	To Adjust To Alterations	Change Requests	As Needed	Meetings, Teams, Jira	Person Requesting Alteration
SW Developers	To Adjust To Alterations	Alteration Requests	As Needed	Meetings, Teams, Jira	Person Requesting Alteration
Project Team	To Sync With Each-other	Work Updates	Daily	Personal Meeting	Project Team
SW And HW Developers	To Fix Problem	Minor Problem	As Needed	Jira	Person Requesting Fix
SW And HW Developers	To Fix Problem	Major Problem	As Needed	Jira, Teams, Personal Meeting	Person Requesting Fix
SW And HW Team Leads	Maintaining Project On Track	Discuss Requirements	As Needed	Direct Communication	Stakeholders

Table 4: The Communication plan for the team.

9 Risk analysis and response planning

From our project planning phase, we collectively conducted a risk analysis described in this section. The risks were identified after the majority of the project plan had been developed, to better understand all the potential risks of the envisioned system. The risks were collectively identified along with our estimation of their probability of occurring and the possible impact on the project. This was performed to highlight sensitive risks to consider, and develop possible mitigation strategies to lessen or avoid their impact during the remaining project phases. The results are outlined in Table 5.

Risk	Probab. (P) 1 to 5	Impact (I) 1 to 5	Risk Level $P \cdot I$	Risk Response
Issues with team member absence, sick leave, etc.	1	3	3	Continuous communication and meetings within the group, ability to cross-train members

Problems in the team, team cooperation, dynamics	2	3	6	Try to resolve eventual issues immediately, ask teachers for help if the issues do not resolve
Issues with work environment (access, computers, internet)	1	3	3	Attempt to work from home, use an alternative work environment with possible remote tasks
Unreasonable project scope	2	4	8	Focus on delivering a minimal viable product before increasing scope
Poor time management / insufficient time	3	4	12	Conduct a thorough project planning phase, continuous follow-up meetings, adapt to changes
Miscommunication / inconsistent information	3	4	12	Conduct daily scrum meetings to assess where we are at, what has been said and done, problems, etc.
Not following UAV/UGV/other legislation	2	2	4	Keep requirements in mind, the drone is not free-flying, do not publish sensitive information, etc.
Component delivery delay	3	4	12	Prepare a plan of items that need to be purchased and order them in time, attempt workaround temporary solutions
Broken/missing components	3	4	12	Attempt to fix the component, find a replacement, or order a new component, plan all components
Insufficient budget	2	4	8	Carefully plan required components and possible backup solutions, attempt to secure additional support
Bad weather/environment conditions	2	2	4	Test in advance, wait for better weather, focus on other tasks meanwhile. Consider small-scale experimental testing at first
Human injury or environmental damage	2	3	6	Simulate and test the systems, run small-scale experiments and increment, ask questions/research if unsure, use the Aj, Oj, Halloj
Insufficient battery life	2	3	6	Accept small scale prototype delivery system, test for sufficient delivery between buildings. Implement low-battery warning
Too heavy cargo or too big payload for the delivery system	2	2	4	Focus semi-autonomous delivery system at first, scale up later with a trailer or other components
Electrical or thermal hazard	3	5	15	Plan for sufficient power consumption, test and verify electrical system to avoid overheating
The drone does not fly/land	3	5	15	Emulate the drone flying through some stationary solution, buy a radio controller as a backup, semi-autonomy, human intervention
Husky cannot drive	3	5	15	If it cannot be fixed within a reasonable time, accept and focus on the drone: vision, anti-collision, autonomy, etc.
Faulty emergency stop	3	4	12	Implement logic to avoid accidental stop, manual human intervention if some catastrophic situation
UAV tether/cables get stuck or loose	2	4	8	Accept experimental setup without obstacles such as trees, birds, etc. Test drone flight. Use human intervention in case of emergency

Incorrect sensor data or poor sensor fusion	3	3	9	Prioritize emergency break to avoid a collision, test the system limitations, weigh reliable sensors higher
Erroneous communication between sub-systems	2	4	8	Prioritize emergency break to avoid a collision, utilize tested frameworks and APIs
Loss of communication between systems	2	5	10	Use keep-alive/hello messages and emergency stop in case communication breaks, use human intervention
Shuttle collision with environment	2	3	6	Simulation, mapping of the environment, verify collision detection, implement emergency break
Poor system navigation capabilities	3	3	9	Test and improve navigation, utilize promising libraries and solutions, accept a semi-autonomous system depending on time restrictions
Software bugs	4	4	16	Thorough testing process, follow coding standards to catch abnormal machine states
Software incompatibilities	4	3	12	Research dependencies/drivers, compatible software, attempt to use and modify ready-made libraries, solutions, and adapt
Outdated SW/HW documentation	3	4	12	Use the available information, check forums/the internet for additional information, avoid too outdated technology
Inability to utilize 360 camera	2	2	4	Utilize the other sensors for navigation, anti-collision, etc. Use 360 camera as an extra complementary video stream

Table 5: Risk analysis, risk levels and responses.

10 Documentation plan

The documentation plan describes the procedural framework for the documentation throughout the project's lifecycle.

10.1 Documentation Process

All code, design-related documents, reports, and manuals will be stored in the project's GitHub repository which all stakeholders will have read access to. During the progress, all Overleaf documents will be stored in Overleaf but once they are reviewed and finalized they will be uploaded to the Git repository in .pdf format. WBS, work packages, project status, Gantt Chart, and other project management-related documentation can be accessed on Jira Software upon request by the stakeholders.
Templates will be developed and distributed to team members to explain how different types of documents should be structured.
Code, design documents, reports, and user manuals will at least be updated during the completion of a work package. Project management-related documentation will at least be updated during daily meetings.

10.2 Document Review Process

Once a document/part of a document is completed by a person it will be reviewed by another person in the team. If possible, the person who reviews the code should be the Hardware Team Leader for hardware-related documentation and the Software Team Leader for software-related documentation. If the documentation is made by any of the Team Leaders the document should be reviewed by another team member. This applies to all types of documentation in the project.

10.3 Responsibilities

- **Hardware Team Leader:** Oversees, approves hardware-related documentation and assigns hardware-related documentation tasks.
- **Software Team Leader:** Oversees, approves, and assigns software-related documentation tasks.
- **Hardware Developer:** Creates and updates hardware-related documentation as assigned.
- **Software Developer:** Creates and updates software-related documentation as assigned.

11 Testing Plan

The requirements need to be evaluated by testing. Some of the general and project requirements cannot be submitted for testing. The procedures for testing some of the general requirements, in addition to testing all functional and non-functional requirements, are described below.

11.1 Testing plan of general requirements

* *The system shall consist of a Husky UGV and a UAV in the form of a quadrocopter drone.* See Section 4.1, item 6.
  - The UAV and UGV will be independently developed and unit-tested. An integration test will be performed on the system as a whole. The main purpose of this integration test is that the UAV follows the UGV, outdoors, on a straight path without any obstacles between 331 and 326 buildings in Finnslätten and delivers the robotic hand as a deliverable package.
* *The UGV shall be equipped with a lidar, and at minimum two cameras.* See Section 4.1, item 7.
  - To test if the lidar connected to PX 2 is functional or not, e.g., through Slamtec RoboStudio[5] software assessing the functionality. When the UGV camera is connected and turned on, the only functionality check is if pictures or videos can be saved on the operating system.
* *The system will have a power connection between the UGV and UAV.* See Section 4.1, item 8.
  - According to the final prototype, the UAV will not have its own battery, it will be powered from a cable attached to a battery on the UGV. The powering system which consists of batteries and cables should be tested before connecting to the UAV. The UAV power consumption will be measured for the UAV's required flight modes (take off, hovering, etc.). The continuous operational consumption of the drone should be simulated for at least 10 minutes. The voltage drop to the consumer and the condition of the converters and cables have to be assessed. To ensure that all the system parts have sufficient power and not going over capacity. After assuring the correct functionality of the powering system independently, the powering system attached to the UAV will be tested under the same conditions.

11.2 Testing plan of functional requirements

1. *The UAV shall provide a live stream from the Ricoh Theta X 360-degree camera to the UGV.*
   - In order to fulfill this requirement the system needs only access to the 360 camera stream. Then the camera will be attached to the drone and tested manually to verify its functionality. It can be tested by, e.g., recording a short video, or taking some pictures and assessing the correctness of the files on the UGV.
2. *The UAV shall also be equipped with a regular camera to locate and track the UGV.*
   - Features such as gain, white balance, focus, etc. that might affect object detection will be considered. Additionally, frame rate and latency will be assessed to ensure that the camera will be able to capture and process frames quickly enough to be appropriate with UGV's speed without any significant latency.
3. *The UAV and UGV shall support two-way communication.*
   - This requirement will be fulfilled by using either a wired solution with Ethernet, serial or a wireless solution such as Bluetooth to communicate between the UGV and the UAV. Note that not all components of the UGV and UAV will be required to communicate, the main requirement here is that simple command and control data can be sent from one system to the other for general movement.
4. *The UAV shall have an emergency stop and landing functionality.*
   - A unit test will be done to verify the functionality of the emergency stop software before deploying it. It should ensure that the stop command is received by the flight controller and that the flight mode is changed to, e.g., loiter, idle, or hold mode. Manual tests in a controlled environment will be performed to test that the stop functionality works. The level of difficulty of these tests can be incremented, e.g., stopping at different altitudes, in the middle of a turn, before colliding with an object, testing the responsiveness of the stop, etc. In case of an emergency situation (e.g., the UAV is tilted above a certain degree, the camera cannot locate the UGV pattern, communication loss, etc.) the stop should also be utilized.
5. *The UAV shall be able to be controlled manually using radio.*

[5] https://www.slamtec.com/robostudio

11.3 Testing plan of non-functional requirements

- Through a direct connection or via radio control the UAV will be tested to understand the required flight parameters to fulfill its mission. Smaller unit tests will be performed to, e.g., safely hover and land the UAV, the UAV should be capable of following some predefined path at a predefined altitude with some predefined speed. The first phase is implementing basic flight capabilities. Afterward, some degree of autonomy will be implemented by incorporating computer vision control. The last phase has been done to ensure that the UAV can safely take off, land, and keep within a two-meter radius of the pattern it is tracking.

6. *The UGV will have a landing pad for the UAV.*
    - After building the landing pad and connection to the UGV, several factors should be checked and considered. First is that the landing pad connection to the UGV is robust and can handle the UAV weight and its landing and taking off functionality. On the other hand, when the robot is moving, this landing pad should not limit the maneuverability of the UGV. Finally, the landing pad should not obstruct any of the Husky's sensors or cameras.
7. *The UGV shall have emergency stop functionality.*
    - After implementation, a unit test will be done before deploying the software on the hardware. The stop signal will be mocked and sent to the function. The function should return correct outputs in order to enable the locks. The stop function should be able to send correct signals to the UAV to stop or land as well. This will be checked based on the interface that sends and receives ROS2 commands between UGV and UAV. The stop software should also be able to send desired signals to shut down the power. In the end, an actual test will be performed with the final product to ensure the correct functionality of the e-stop software.
8. *The UGV shall be able to access data from the extra-equipped sensors.*
    - Based on the sensor type being used, the sensor output value can be printed to the terminal or saved in a file.
9. *The UGV shall be able to be directly controlled via a gamepad.*
    - This functionality will be deployed as a function, all the inputs as buttons that were planned to have a functionality should be mocked and tested according to the functionality. Each input should enable a predefined function. This will be tested by assuring that the correct ROS command is being transmitted. Finally, the controller will be connected and the buttons will be pressed several times to ensure that the UGV will follow the correct behavior.

11.3 Testing plan of non-functional requirements

1. *The UAV shall autonomously follow the UGV, staying centered above it within a two-meter radius.*
    - The computer vision software will be tested and optimized in order to fulfill the requirement. The UGV has a top speed of 1 m/s and it will have a pre-defined pattern attached to its landing pad. The UAV will be following this pattern through its vision algorithm, i.e., tracking the center of the pattern and adjusting accordingly. Therefore, tests will be performed to evaluate the computer vision's capability of finding the center of the pattern in some time span through software.
2. *The UAV shall be tethered to the UGV using a wire with a maximum length of five meters. Other cables shall have a maximum length of 25 cm less than the tether.*
    - A sturdy rope or lightweight metal wire will be tested so that the UAV's maximum upward throttle cannot break the tether. Manual testing of pulling the tether will be performed. Some safety margin will be added to ensure it can support some extra tension caused by the environment such as moderate breezes (< 8 m/s). The connection tether will be measured to be shorter than any other power or communication cable. The metal cable will be measured to be 25 cm shorter than other cables.
3. *The system shall be able to make one 25 0m trip on a flat surface.*
    - The UAV and UGV will be independently tested to verify that they have enough power to last for a 25 0m trip (the approximate length between buildings 331 and 326) before being integrated as one system. Only flat surface, asphalt or indoor flooring is considered, without aerial obstacles such as trees, birds, etc.
4. *The UGV and the UAV shall manage 10 minutes of continuous operation.*
    - Both separate unit tests and a system integration test will be performed in order to confirm that the system has sufficient power. Both UGV and UAV will be run at maximum power for more than 10 minutes to ensure unexpected events that extend the final performance time.
5. *The UGV shall have a loading area capable of carrying a maximum of 40kg.*

- Based on the UGV's datasheet, it can carry 75 kg of payload. In this project, the Husky will be equipped with various sensors and a new on-board computer. With this equipment in mind, the payload has been adjusted. Furthermore, the new loading area will have a different payload tolerance than the Husky itself. The loading area will be tested by adding weight to it and the stress, strain, or failure of the robot will be monitored.

6. *The system shall have an emergency stop so that both vehicles stop within 500 ms (UGV stops moving, UAV hovers in place).*

   - By using a video recording of when the emergency stop command was sent to the system we can measure that this requirement will be fulfilled approximately.

7. *The UAV motors and power system shall allow a minimum lifting capacity of the UAV itself and its connected cables.*

   - The lifting capacity can be tested using a radio control with a battery on the UAV in a free-flying manner. The lifting capacity can also be tested with power and communication cables directly attached to the UAV. For testing the capacity exact incremental weight will be attached to the drone.

12 Handover plan

Hardware components will be transferred to the stakeholder Martin Ekström at the end date of the project course, 2024-01-12 at C2 Building 331 Finnslätten, Västerås. Our handover methodology includes presentation, demonstration, and documentation of the whole system. Documentation can be accessed from the project's GitHub repository and will consist of code, design documents, reports, inventory of items used in the project, and user manuals.

13 Individual contributions

Almost every topic and section in the project plan has been a collaborative contribution. However, the main responsible person for each section and extra contributors are listed in Table 6.

Assignment	Main responsible	Contributor
Project Information	Sharifeh Yaghoobi	Andreas Johansson, Elon Pettersson
Purpose and Goal	Erik Rågberger	Andreas Johansson, Walter Lagerhäll
Project and Product Requirements	Andreas Johansson	Sebastian Leclerc, Kasra Ekrad, Sharifeh Yaghoobi
Development Plan	Kasra Ekrad	Sebastian Leclerc, Sharifeh Yaghoobi
Documentation Plan	Walter Lagerhäll	Andreas Johansson
Testing Plan	Sebastian Leclerc	Kasra Ekrad, Sharifeh Yaghoobi, Andreas Johansson
Risk Analysis and Counteractions	Sebastian Leclerc	Walter Lagerhäll
Work Breakdown Structure	Andreas Johansson	Everyone
Communication Plan	Elon Pettersson	Andreas Johansson, Walter Lagerhäll
Project Schedule	Andreas Johansson	Walter Lagerhäll
Limitations	Walter Lagerhäll	Andreas Johansson
Handover Plan	Walter Lagerhäll	Andreas Johansson

Table 6: Individual contributions

References

[1] Nyit Sin Phang. Tethered operation of autonomous aerial vehicles to provide extended field of view for autonomous ground vehicles. (2006):64.

[2] Takahiro Miki, Petr Khrapchenkov, and Koichi Hori. UAV/UGV autonomous cooperation: UAV assists UGV to climb a cliff by attaching a tether. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8041–8047. ISSN: 2577-087X.

[3] Seiga Kiribayashi, Kaede Yakushigawa, and Keiji Nagatani. Design and development of tether-powered multirotor micro unmanned aerial vehicle system for remote-controlled construction machine. In Marco Hutter and Roland Siegwart, editors, *Field and Service Robotics*, Springer Proceedings in Advanced Robotics, pages 637–648. Springer International Publishing.

[4] Christos Papachristos and Anthony Tzes. The power-tethered UAV-UGV team: A collaborative strategy for navigation in partially-mapped environments. In *22nd Mediterranean Conference on Control and Automation*, pages 1153–1158.

[5] Miguel Nakajima Marques, Sandro Augusto Magalhães, Filipe Neves Dos Santos, and Hélio Sousa Mendonça. Tethered unmanned aerial vehicles—a systematic review. 12(4):117.