Justification document

Thomas Schenk

Contents

[Introduction 3](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1277_3812656425)

[1. Proof of concept Map environmental structures with LIDAR 4](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1279_3812656425)

[Context 4](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1281_3812656425)

[Problem/Opportunity 4](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1283_3812656425)

[Goal 4](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1285_3812656425)

[Process 4](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1287_3812656425)

[Validation 4](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1289_3812656425)

[2. Integration of a new camera 6](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1291_3812656425)

[Context 6](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1293_3812656425)

[Problem/opportunity 6](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1295_3812656425)

[Process 6](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1297_3812656425)

[Validation 6](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1299_3812656425)

[3. Project Plan Personal project 7](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1301_3812656425)

[Context 7](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1303_3812656425)

[Problem/Opportunity 7](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1305_3812656425)

[Goal 7](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1307_3812656425)

[Process 7](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1309_3812656425)

[Validation 7](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1311_3812656425)

[4. Better Autonomous driving 8](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1313_3812656425)

[Problem 8](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1315_3812656425)

[Process 8](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1317_3812656425)

[Validation 8](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1319_3812656425)

[5. System Integration of Object Detection 10](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1321_3812656425)

[Context 10](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1323_3812656425)

[Problem 10](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc589_3853979228)

[Goal 10](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc591_3853979228)

[Process 10](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1325_3812656425)

[Validation 11](file:///C:\Users\Thomas\Downloads\Justification_document.docx#__RefHeading___Toc1327_3812656425)

## Introduction

In this document you will find the justification of each piece of work done by me. Some parts are already done, while some problems will reiterate a few times throughout the sprint, depending on their size.

## Proof of concept Map environmental structures with LIDAR

### Context

The JACKAL is aware of its surroundings. A key part of this is being able to recognize and see certain obstacles. This requires detailed sensors to provide the JACKAL with qualitative data to make these recognitions. But not only the JACKAL needs qualitative data to understand its environment, it also needs this data to be useful enough to serve to the end user so that it can understand the environment as well. The end-user could make use of this data in the form of a map or a 3d visualization.

### Problem/Opportunity

Currently the data provided by the JACKAL is inferior to the data which could be provided if the hardware would be upgraded. The current data which is provided by the robot is not good enough. The 2D map, although understandable, would benefit massively from a new LIDAR which is a direct hardware upgrade. And the 3D mapping, currently done by a camera, is almost too detailed, showing way too much data of the visuals around it; creating a very fuzzy and illogical image.

### Goal

Create a proof of concept which shows that a 3D LIDAR is a viable method to upgrade not only the 2D but also the 3D map by providing more accurate and quantitative data to construct these maps. The end-user benefits massively by having a map based on more and more accurate data to gain environmental awareness, this proof of concept aims to provide an answer to the question whether it is possible to create this data with the use of a new 3D LIDAR.

### Process

To start this proof of concept I first looked at viable algorithms to support the new LIDAR. Since the LIDAR natively has a high tunability, we were looking for an algorithm which can incorporate this. After finding one I integrated the new algorithm on my computer and connected my LIDAR.

When I made sure this was working, I went ahead and connected everything to the robot. I did this intuitively since the algorithm used is supposed to be a lightweight package. Therefore, I was curious as to how it would perform on a system which is arguably slower and has more latency.

In order to serve the same data but via the robot, some additional infrastructure had to be set up to accommodate this. After integrating the LIDAR and replacing the old algorithm with the new one; I was done.

### Validation

The validation of this proof of concept in its core are very simple, the visualization needs to be “good” enough to see what is going on. However, what is good? I determined that being able to quickly recognize a room and its contents is a must for this project. The key objects in the room should always include the walls and passageways/doorways. Pillars, poles and other, smaller, vertical objects should appear on the map as measured data.

To combat pattern recognition while testing this “room recognition”, I drove the robot to a part of the building which I had never seen before. Letting the robot scan the surroundings for a while; I then went back to see if what the robot saw was how I envisioned the room. These results seemed to overlap quite nicely.

I briefly mentioned the efficient speed of the algorithm, inherently this means that speed is desirable in this application. To test the speed correctly, I needed to recreate the scenario like it will be when the end users use this product. Which is remotely via some sort of data tunnel. In my case, SSH is the option I went with. I tested the speed of the feed by having colleagues walk and perform quick jerky motions in front of the robot. If I could distinguish in a reasonable reaction time if there was a person or what they were doing; the speed would be adequate. This happened to be the case.

## Integration of a new camera

### Context

We would like to make the JACKAL more aware of its surroundings, while simultaneously make it easier for the end user to see what the robot sees. Some data is interpretable way quicker for a computer than a human, for example highly noisy point clouds. This is currently solved by a camera at the front of the robot, in its current configuration it is responsible for 3D scanning; as well as deliver a camera feed to the end user if desired.

### Problem/opportunity

The current problem is that the camera feed generated by the camera is very slow/choppy. This feed is not reliable at all for first person driving or accurate mapping. Let alone being able to monitor or recognize anything going on when far away from the robot. Furthermore the PO gave us a requirement which requires the robot to be able to see in the dark, the current camera did not support IR camera modes and therefore does not honour the requirements.

### Process

First, I made sure the camera worked according to the factory standards. I did this by installing the manufacturers software and checking all output feeds. After this, I researched how the current camera on the jackal actually talks to the rest of the system. This would be my beginning point of integration, since the system should start to listen to a new camera.

When I was done with this, I made sure the camera also worked in the ROS configuration on the robot; by running it locally on my pc with the same settings. The feed in this preliminary testing was very good and clear, so I was quite opportunistic to integrate it.

After this, I integrated the camera in the system. First of course all the required drivers, then I made the new camera talk to the existing system. Because I had first figured out how the previous camera worked, this was a very smooth job to do. I had to alter a few lines of code in setup files to accommodate for the new drivers.

### Validation

To validate this outcome, we can simply view the infrared camera stream and see if it outputs something which is not a black screen. This happened to be the case. The requirement was fulfilled.

We also wanted to fix the latency issues. To test the speed correctly, I needed to recreate the scenario like it will be when the end users use this product. Which is remotely via some sort of data tunnel. In my case, SSH is the option I went with. I tested the speed of the feed by having colleagues walk and perform quick jerky motions in front of the robot. If I could distinguish in a reasonable reaction time if there was a person or what they were doing; the speed would be adequate. This happened to NOT be the case. Sadly, the choppy feed was not due to the internal speed of the camera.

When testing the robot for its primary application, driving from first person view. The feed sadly wasn’t any better. Though it was actually doable to drive it, with little pauses in between it was no problem to navigate through the building.

## Project Plan Personal project

### Context

For my Personal Project I need to keep track of when I am going to do what and what my steps are. This way I can better plan the project, as well as project it to others a lot better.

### Problem/Opportunity

With this projectplan I could improve the way I manage and present my project better to others.

### Goal

Have a project plan which lists the scope, steps needed, the planning and a risk analysis.

### Process

I sat down and made sure to write down everything I knew about the project on a piece of paper. Then I slowly made it into separate boxes so I could plan and phase it better. Then it was a matter of making it into a project plan.

### Validation

I think the only possible validation for this is feedback from the teacher. Furthermore a good validation can be at the very end of the project. Did the project plan fit the phasing of the project? The better this fits, the better the project plan was.

## 4. Better Autonomous driving

Context

The JACKAL is able to drive autonomously. With the old sensory it managed to traverse buildings. With the new and improved hardware, we aim to not only replicate this; but also improve where possible on this feature.

### Problem

The current configuration on the robot still assumes old hardware is in place. As we have seen before, new hardware is present on the robot to improve many things; amongst which is autonomous driving. The robot currently does not know however, that this hardware is there and still assumes old hardware is in place. Because of this, the robot cannot drive autonomously. Due to this being a major part and requirement of the robot, we need to rewrite the robot to accommodate for this change; which in turn provides us with the correct basis to re-enable autonomous driving.

### Process

When installing the hardware required for this change a sprint back, I gained quite a bit of knowledge where ‘everything was’ already. This makes navigating through the files and making changes a breeze. The actual part of setting up the required code therefore was done very fast.

What came less fast, were the bug fixes which needed to be done. Namely one bug, in the local costmap. This took me very VERY long to figure out, since I had to retrace various topic connections and links before I even understood what was going on.

Finally, the problem was found and I was able to fix the blocking bug. Which enabled this product to be finished.

### Validation

Validation of this project came in three stages:

– No autonomous driving  
This was very easy, no autonomous driving; problem is still there = problem is not solved.

– Autonomous driving, but many collisions  
At this point the problem was half tackled. The robot was able to drive on its own again, but crashed into everything in sight. Or well, for the robot not in sight. While technically the robot DID drive autonomously, a drive with a crash isnt very beneficial to the robot or its intented purpose. This means the problem still is not solved.

– Autonomous driving, without collisions

At this stage, the problem was tackled. Which could obviously easily be recognized by the fact that the robot was comfortably exploring and driving around the room. Seeming very smooth and steady. Finally, the problem was solved.

## 5. System Integration of Object Detection

### Context

To determine points of interests (POI) on the map, the JACKAL makes use of the installed camera to detect objects of interest. It does this using a recognition model using python. The operator can see these points of interest on the map and inform others around him.

### Problem

While the code for object detection is already present, the integration with the exploration package is very poor/nonexistant. This integration essentially contains a launch file which launches everything at once. The JACKAL currently does not have enough processing power to accommodate for both these high intensity programs at once. This causes the JACKAL to lag tremendously. Indirectly this causes both the exploration and the object detection processes to not perform, or perform very poorly.

### Goal

Create a different system architecture where computational loads are divided or made more efficient in order to not impede upon the quality of already well performing processes. After this has been established, make sure the object detection can be ran parallel to the exploration process in this new architecture.

### Process

To start this problem off, I first looked carefully at the problem; and what its final solution should contain. I considered ease of use, technical complexity but also ofcourse: what is this solution going to mean for the load on my robot?

After carefull consideration, I determined that the best way to split up computational loads; is to introduce a new computational device to the already setup ROS network. Which is setup by the JACKAL. This brought a new question to the table: How?

Again, after carefull consideration of how to best connect these now two devices; I settled on a physical link between the master (robot) and the slave (extra device). I made a new, more inherent to its name, script to quickly connect these two devices.

After this, I needed to run the object detection on the extra device; and the exploration package on the robot. Which proved to work near flawlessly; with the only exception to this being a bug in the part which marks the found object on the map.

### Validation

Validation for this problem can be split into two categories.

– New architecture  
To validate if the new architecture works, or in other words: are these two ROS systems connected?

We can simply run a ROS command which enables us to deduct this. This command is *rostopic list,* which lists all the topics a ROS machine can see. If we run this on the extra device and see topics such as: /scan, /move\_base or /velodyne; we know for sure these two systems are connected. Since these topics are ONLY published and made by the robot. Viewing them on our extra device, means we got a connection!

– Object Detection  
With the absence of the marker previously mentioned, how can we possibly determine that objects are detected?   
Luckily, this object detection does not only show its succes (or failure) by placing markers. Every object that is detected comes with a picture of that specific object. On this picture the model draws a border around the object it has detected. These are stored in a folder on the host machine, which we can check. If there are pictures in there; we know it works!

## 7. Dashboarding ROS data

### Context

The Jackal is a product that will be used in high intensity situations, or very low intensity situations. Either way, it needs to have an intuitive and clear user interface. Without it, its functionalities will be forever lost in the complexity of separate terminals and different visualization programs.

### Problem

The Jackal currently has no 1 place where the end user can both see and control the data on the Jackal. To be effectively rolled out as a product; it needs this. The operator in the field cannot be expected to carry a laptop and juggle around CLI (Command Line Interface, otherwise known as Terminal) windows or other programs currently needed for executing certain behavioural actions on the Jackal.

### Goal

Something needs to be created to eliminate the need for multiple instances of programs, in whichever way shape or form, and which serves the end user with one concise GUI. The form of this product is very free, yet it must have one specific attribute: it must be able to do everything on one page.

Process

The first task was figuring out what type of program would be used to display everything. While an application on a desktop (like how you would launch Word) would ultimately be the most professional. We opted for a web based approach. Most notably for its already existing (albeit very vague) ability to interface with remote Ros environments. Furthermore, web applications are easy to port into windows applications; making this work non-redundant if in the future a port like that is preferred.

After this, implementing Ros web connectivity was achieved by using ‘rosbridge’. A service which provides a WebSocket. Since our machine uses a vpn, this was a bit of a different setup than usual; but it proved to work.

After creating a websocket, specialized javascript libraries were used in order to create visualizations and user interactions.

Validation

Validation for this product is very easy, it has a set amount of requirements; all of which can be individually met or not. In our case, the current requirements are met.

1, Live video feed

2, Live Map feed  
3, Live Exploration control