Tooltracker
Final Report

Mechatronics, Advanced Course
MF2059
January 15, 2016

Written by

Satyajit Bagchi	sbagchi@kth.se
Adam Ekström	adameks@kth.se
Christoffer Lindahl	clinda@kth.se
Tobias Lundin	tolundi@kth.se
Lin Xuan Lyu	lxlyu@kth.se
Björn Magnusson	bjomag@kth.se
Mattias Nilsson	mattiani@kth.se
Elin Nordmark	emkno@kth.se

The Royal Institute of Technology
2016 KTH
Stockholm

Abstract

This report investigates the possibility to create an indoor positioning system for an industrial environment. The research project is conducted as a joint project together with the external client Atlas Copco. The research method has been to first conduct a deep-scan of current state of the art solutions and techniques for indoor positioning systems. Second, evaluate three selected techniques against the requirements set together with Atlas Copco in order to decide an final system design. Third, a testing platform for sensor fusion and the individual techniques were built.

In conclusion, it is possible to build a solution where different techniques are fused together to overcome the individual weaknesses within each technique. However, it was not possible to solve all the requirements in all situations within the scope of this project. The results shows that the proof-of-concept prototype are able to meet the requirements within line of sight and in slightly covered line of sight conditions, although, sufficient accuracy was not achieved in non line of sight. The requirements on timing and update frequency was met in all situations.

Contents

Abbreviations

AC	Alternating Current
AoA	Angle of Arrival
BRS	Bolt Recognition System
CAD	Computer-aided Design
DOF	Degrees of Freedom
EKF	Extended Kalman Filter
GUI	Graphical User Interface
IMU	Inertial Measurement Unit
IR	Infra Red
LED	Light Emitting Diode
LOS	Line of Sight
NLOS	Non-Line of Sight
PC	Personal Computer
PCB	Printed Circuit Board
SV	Stereo Vision
SVN	Apache Subversion
TDOA	Time Difference of Arrival
ToF	Time of Flight
UDP	User Datagram Protocol
UKF	Unscented Kalman Filter
USB	Universal Serial Bus
UWB	Ultra Wide Band

1 Introduction

1.1 Background and Objective

The company Atlas Copco is a tool developer for industrial companies. One of their areas is the development of hand held tools for industries (such as nut-runners). They have both cord- and battery driven hand held tools. One problems with hand held tools is that the costumers may lose or misplace their battery wireless tools. Atlas Copco already have a tracking system that deals with this problem. Atlas Copco seeks to further develop tracking or localisation systems. They also seek to solve the problem of how for the customer to guarantee that the manufacturing was correctly done. One way to do this is to be able to correctly track what bolts have been tighten and in what order. When a system can do this, it becomes easier to verify that the right torque was used on the right bolt. This means that the system needs to be able to track the nut-runners outgoing taps position accurately down to 5 cm. To be able to find out if this is possible they have turned to the Mechatronics Department of KTH (Royal Institute of Technology). The task was then given to the students reading the Advanced Course in Mechatronics. During the spring of 2015 the students did a state of the art report on what techniques there is to track a object in an indoor environment. One of the problem is the need to track the object when the measurement systems looses line of sight to the object. When the object (in this case a nut-runner) is moved inside a metal casing (to tighten a nut). The state of the art report showed that there was not one single system that could fulfil all requirements.

The solution will be to combine some of the current techniques and make a system that is better than each of the single techniques.

1.2 Requirements

The requirements are divided in two different parts, functional- and soft requirements. The functional requirements can be tested in the prototype by test-cases. The soft requirement can not be tested (or have not been tested for), but the prototype have been designed with the soft requirements in mind.

1.2.1 Functional Requirements

Accuracy:

- The system shall be able to measure the central position of the outgoing tap of the tool with accuracy of 50 mm when at a distance of 600 mm or closer to the target.
- The system shall be able to measure the central position of the outgoing tap of the tool with an accuracy of 500 mm inside a work space.

Output:

- The system shall update the position of the tool with a frequency of at least 10 Hz.
- The system shall output the position of the tool in the form of a Cartesian coordinate system in three-dimensional space (X, Y, Z) and the normal vector of the outgoing tap.
- The origin of the Cartesian coordinate system shall be defined as a fixed point in the work space.
- The systems position lag shall not be greater than average human reaction time, estimated to 200 milliseconds.

Robustness:

- The system shall be robust enough to operate in an simulated[1] industrial environment with a limited line of sight[2] without failing the accuracy requirements.

1.2.2 Soft Requirements

Accuracy:

- The system shall be able to measure the central position of the outgoing tap of the tool with an accuracy of 2000 mm when located in a global indoor positioning system.

Intrusiveness:

- The worker should not feel supervised and/or tracked.

Robustness:

[1] A lab created approximation of an industrial environment containing metal objects and including disturbances from the tool itself.

[2] Objects that partly blocks the line-of-sight between tool and receiver.

- The system should be robust enough to not create a stop in the production.
- The system should not be manipulable by the operator.

Cost: The prototype budget shall not exceed 50 000 SEK.

Scalability: The system should be easy to install/remove/change.

Dimension: A possible add-on to the tool should not affect the usage significantly. The size and weight of the add-on should preferably be of similar dimensions as the current add-ons.

1.3 Scope and Limitations

This report is a result of the KTH course MF2059 Mechatronics, Advanced Course, 15 credits. The main content of the course is the focus on product development of mechatronic product, in large project where teamwork and collaboration with industrial representatives is a core part. 15 credits corresponds to 400 hours work during one semester of 20 weeks. The work starts in week 36, and the prototype have to bee done on the week 50. The semester has two exam-weeks which leaves 13 weeks of work to produce the prototype tracking system, this corresponds to working 30 hours a week, however, calculating that this should be worked on half-time, the result is 20 hours per week. The budget of this project is 50 000 SEK.

The tracking system will be a prof of concept prototype. It is designed to be modular, the individual measurement system can be exchanged with another system (or similar individual technique), with minor changes to the interface. Together these techniques should be able to fulfil the requirements. No wireless communication to the tool will be implemented for this prototype and the sensor fusion used to combine the systems will be perfected to whats within the resources to do. The testing of the system is done on a test rig that simulate a industrial environment. The test rig is used in an room that multiple projects have access to, where all the teams are sharing space and resources. The testing was done in an limited time, since there was only a short time where the group had access to the area/systems.

The deliverable to Atlas Copco is the report, the tracking system software, test rig and three of the measurement systems. The deliverable to the teaching team is the report and a demonstration of the system.

2 Method and Design Process

This section will describe the method used to solve the problem stated in the chapter1.2 Requirements and describe some of the major design decisions taken on system level. First, general research was conducted and summarised in a pre-study report covering the current state of the art techniques available to solve the requirements. Although, no single technique alone could solve all requirements at the same time. Second, concepts were generated that incorporated combinations of four of these techniques in order to solve the requirements at the same time. Third, a final system design was chosen and fourth, the final system were tested against the requirements.

The workflow of the project can be seen in the three separate project plans used during the project. The pre-study was made during the spring semester and the project continued during the fall. During the spring, the work was divided into three phases as seen in Figure 1 below in order to cover an sufficient amount of research for the pre-study. The conclusions drawn from the Pre-study will be covered in chapter 2.1 Pre-Study Conclusion below and the whole pre-study can be viewed in Appendix A.

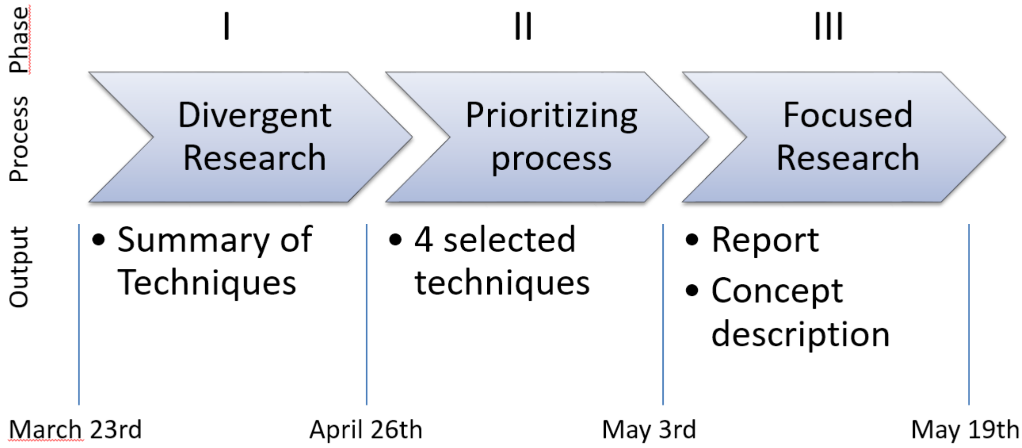


Figure 1: Project plan for the spring.

During the fall the project was split up in four steps, as seen in Figure 2. A detailed Gantt-chart was made to make sure that all dependencies between the different tasks for the fall was understood and also to gain a better overview, the Gantt-chart (which is split up on six pages) can be seen in Appendix C.

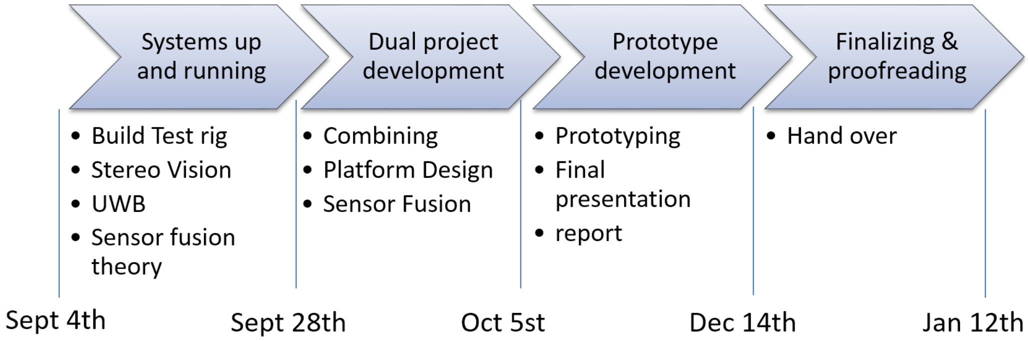


Figure 2: Project plan for the fall.

The final period of the fall was planned more in detail at mid-term to prioritize among the remaining tasks and to verify that all important steps were covered. This plan can be seen in Figure 3.

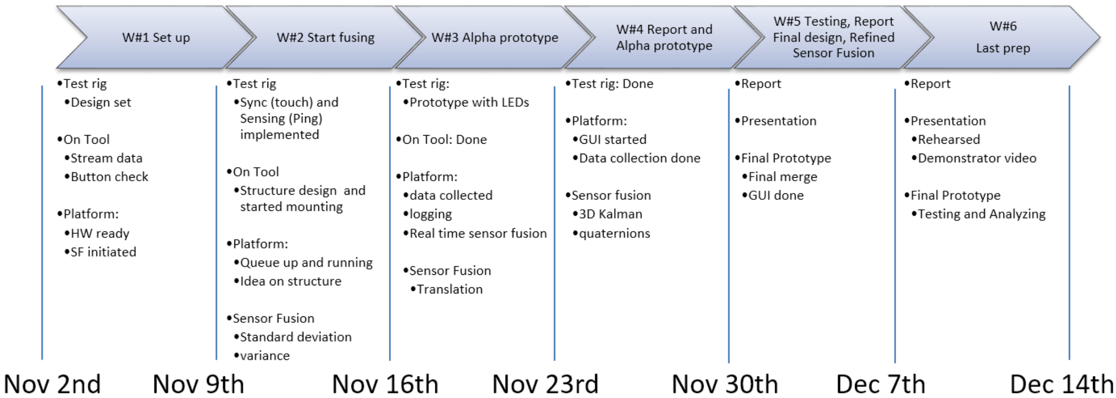


Figure 3: Detailed project plan for the second part of the fall.

2.1 Pre-Study Conclusion

No techniques were able to solve all the requirements at the same time. For instance, stereo vision (SV) system has the drawback of line of sight needed, which cannot satisfy the hard requirement of robustness, see section 2.1.2 Stereo Vision in the pre-study report; Appendix A. Projected IR (infra red) light would be too expensive to implement within the scope of this project. For the purpose of solving all the requirements, the combination of different techniques mentioned in the Vision chapter is researched, see section 2.1 in the pre-study report; Appendix A.

In the Sound chapter, sensor fusion is mentioned as a possible future for the problem of indoor positioning, see chapter 2.3 in the pre-study report; Appendix A. What's more, based on the simulations, it was concluded in Sillero's master thesis that the sensor fusion algorithms significantly improved the performance over single technique, see chapter 2.7.1 in the pre-study report; Appendix A.

Due to the conclusions from the state of the art evaluation, there is a high possibility that the project would fail if the accuracy of 10 mm is kept. As a result, the accuracy is changed to 50 mm. An accuracy of 50 mm is still a valuable outcome for Atlas Copco and achievable within the scope of this project, see chapter 3.4 in the pre-study in Appendix A.

In Figure 4 the accuracy of the individual techniques can be seen, the Triangulation Laser is the most accurate technique, followed by FARO and ToF Laser. However, triangulation laser and FARO are both very expensive. Time of flight laser only has one dimensional output. When the aim of the accuracy is 50 mm, almost every technique except inertial measurement unit (IMU) can be selected to use. Although IMU performs worse in accuracy than other techniques it is the only technique that can completely handle non line of sight, which is one of the requirements.

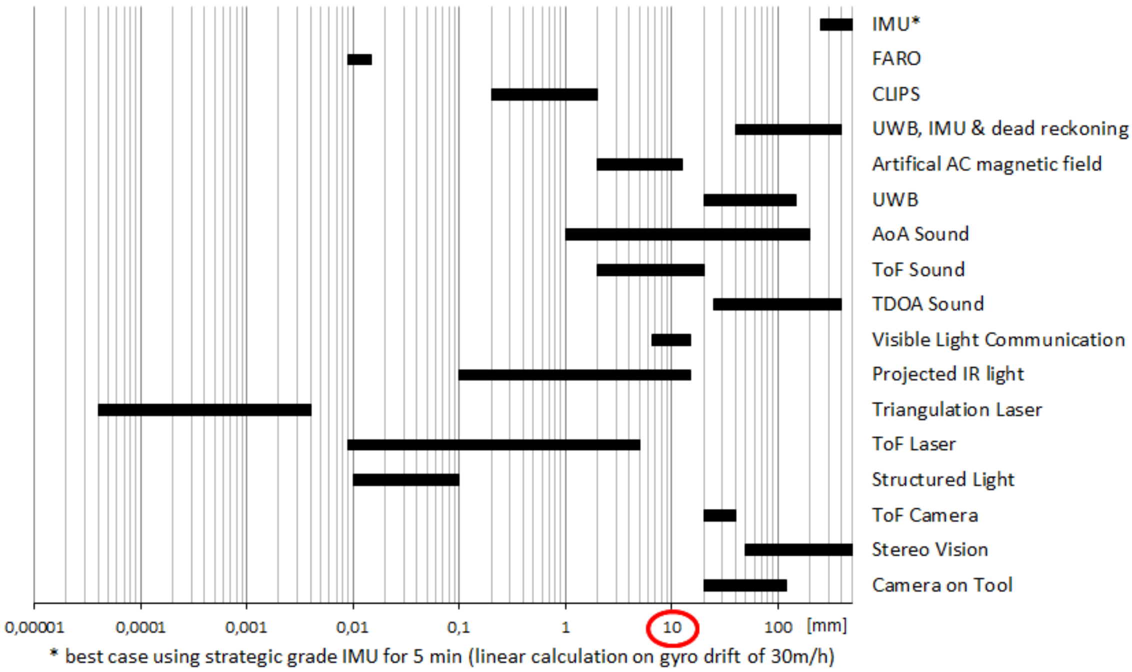


Figure 4: Accuracy for the techniques covered in the pre-study.

Based on the knowledge and confidence, different techniques have been scored by the authors of the report. As can be seen from Figure 5, the promising techniques according to the high grading of the authors of the report are Ultra Wide Band, ToF Sound, stereo vision (SV) and Triangulation/AoA Sound. The belief map summary also shows that FARO, Artificial AC magnetic field and Projected IR light have high scores in confidence, but due to the lack of knowledge and e.g. specific disturbances, these should not be chosen for the project. IMU, as mentioned above, can be used as a complementary technique to deal with the non line of sight. In conclusion, Ultra Wide Band, ToF sound or Triangulation/AoA Sound, stereo vision and IMU will be utilised to form the concept design.

From the pre-study, the individual techniques was re-evaluated with the rewrites of the requirements in mind and compared for the all over design desicion (see Appendix E.1). Some promising techniques was discarded because they could not fulfil the robustness requirement; Visual light communication and Magnetic field localisation. The most promising techniques left was ultrasound (with different or hybrid ways of calculating position considered as one individual technique), Ultra Wide Band (UWB), WiFi localisation, Inertial measurement unit (IMU), stereo vision (SV) and projected IR-light.

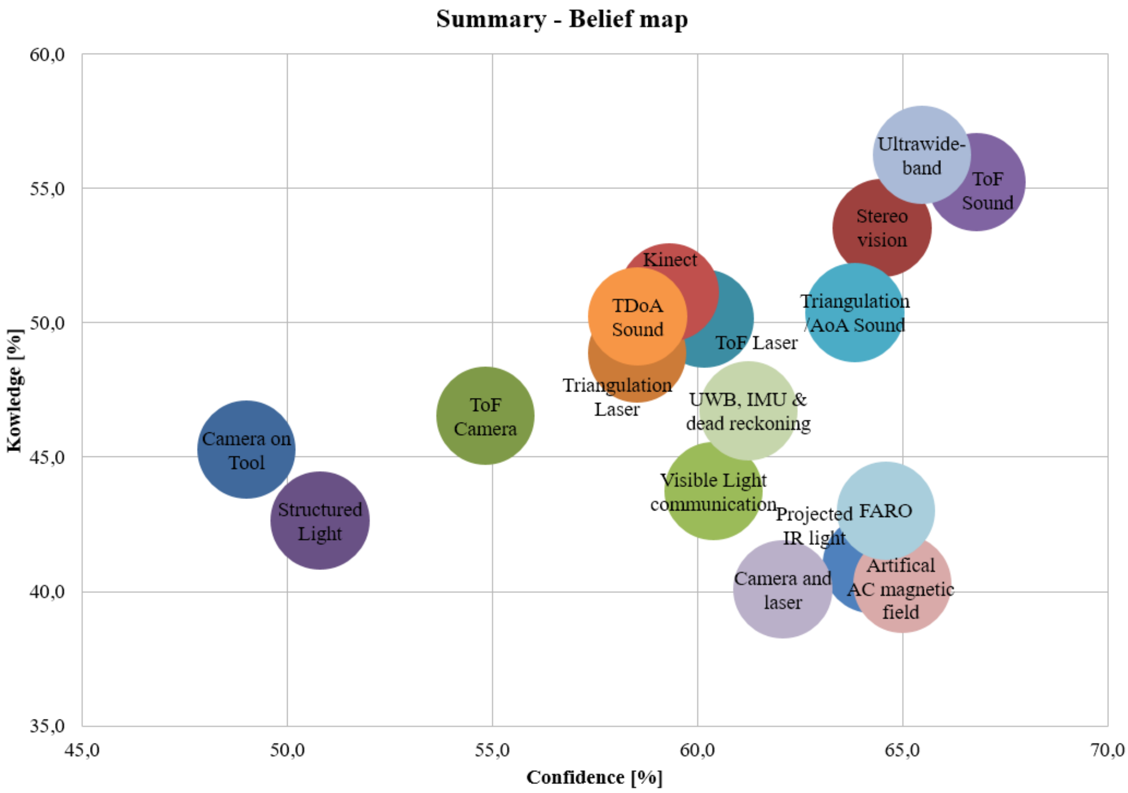


Figure 5: Summary of the Belief map of techniques covered in the pre-study

2.2 Concept Design

Two concepts was developed in the beginning of the project (Appendix E.1), projected IR-light was discarded because the evaluating opinion was that a stereo vision system could probably do the same things but with higher reliability (it was also deemed too expensive, see previous section), and WiFi localisation was discarded because the evaluating opinion was that a UWB system could probably do the same things but with higher reliability. The natural concepts that emerged was therefore based on utilising an inertial measurement unit (IMU), Ultra Wide Band (UWB), ultrasound and stereo vision (SV). Concept 1 entailed utilising inertial measurement unit (IMU), Ultra Wide Band (UWB) and stereo vision (SV), and Concept 2 entailed utilising IMU, Ultra Wide Band and ultrasound. Both concepts would use sensor fusion in order to combine the individual techniques (see 3 Sensor Fusion).

Concept 1 was chosen because the Ultra Wide Band can be utilised to make the global positioning and can handle partial non line of sight (can penetrate some materials), meanwhile a stereo vision system will deliver an adequate accurate local positioning and IMU is for the non line of sight which only this individual technique can handle fully. The UWB can, however, help with the drifting properties of the IMU, and help reset it in partial non line of sight. See the pre-study, Appendix A for further explanations.

Concept 2 entails sensor fusion of IMU, Ultra Wide Band and ultrasound. This solution was chosen because the global positioning is achieved by the Ultra Wide Band and IMU will handle non line of sight positioning the same way as in Concept 1. However, the requirement of accurate local positioning would be met by the ultrasound (see the pre-study, Appendix A for further explanations).

2.3 Individual Techniques

From the state of the art report four indoor localisation techniques was considered (see previous section 2.2 Concept Design), all the techniques have it's own advantages and disadvantages. The decision was made not to build one from scratch, but use existing systems because these systems would give higher accuracy than could be given within the time- and knowledge resources of this project, see Appendix: E.2. Each of the techniques was searched for, until a few suitable commercial systems was found. The systems found are:

- VICON - stereo vision system lent by the department
- TREK1000 EVB1000 evaluation kits - Ultra Wide Band system
- HX19T12 Hexamite evaluation pack 2 - ultrasound system
- MPU9060, MPU6050, and ADIS16445/PCBZ; ADIS1644X/BRACKET; EVAL-ADISZ - IMU

For each of the techniques, the system was acquired, but the ultrasound system was delayed. When the ultrasound system finally arrived, the time was deemed to be too short (see Appendix: E.5) left on the project to focus on getting ultrasound up and running.

2.4 Final System Design

Because of the tardiness of the delivery of the ultrasound system the efforts were instead focused on the three other systems, and therefore Concept 1 (see section 2.2 Concept Design) is the only concept built. The design of the final system is to combine several positioning systems to create a more reliable and accurate one using sensor fusion. The final system is designed to be modular, see Appendix E.3, so each of the subsystems can be replaced with an other similar tracking system. This was decided at the second Design review with the teachers because the knowledge and time resources was not enough to make the sensor fusion impeccable. There would be no time to test and analyse that the most effective sensor fusion is implemented, nor is this kind of data analysis within the general knowledge of a mechatronics engineer. To give more value to the company (Atlas Copco), it was argued that a platform for doing sensor fusion on would give a reasonable proof of concept adding value to their own development. They could also use the platform to test and compare different techniques. The deliverable is therefore a platform where sensor fusion between different indoor localisation systems can be tested and verified. This final design decision was taken together by the students (the authors of this report) and Atlas Copco, see Appendix E.5.

The platform architecture and build up was discussed continuously throughout the design process. Initially, Matlab seemed as the natural choice as the team members had all extensive knowledge of the software. However, this was discarded as there was a fear that the sensor fusion might react negatively to that level of abstraction (introducing unknown delays) and also because the company, Atlas Copco, should not need to acquire such expensive licences to use the system. Therefore was Matlab merely used throughout the project as a tool for verification.

The hardware of the platform was discussed with the PhD student Sagar Behere, see Appendix E.4. Many options was discussed, the decision is generally a trade-off between computation power and how close to real time the system should be. Since the sensor fusion should be possible to extend and also to support a GUI, one of the more computational powerful options was chosen. The platform also need to be connected to other systems so therefore a PC was the best option, this would facilitate communication protocols and such since there are many open source resources to be found online. Linux was chosen because it can in some ways can give real time with preemptive options for its processing which could be used if the sensor fusion showed symptoms of suffering from too large delays (sensor fusion is often a discrete time-dependent algorithm).

2.5 Testing Method

The goal of the project is to generate a system that will combine several different measurement systems, to create a better system that can perform task that the individual subsystem can not do. The requirements have been set up so define this system, the aim is to test the system against the functional requirements. The accuracy, output and robustness requirements must be fulfilled for the improved system to be valid. The accuracy is tested by measuring the difference in position in a Cartesian coordinate system. The lag in the system is tested by measuring the difference in time. The robustness is tested with the test rig that simulates an industrial environment.

A template for recording testing has been created, where the collected data is recorded and statistical analysis of the data is performed. The standard deviation and variance is calculated from the data set. The sample size of the tests was decided to be the quantity of 100 due to limitations (see 1.3 Scope and Limitations).

2.5.1 Accuracy and Timing

Accuracy is in this case the difference between the actual and calculated position of the outgoing tap of the tool. The accuracy requirements are divided in two parts. The first is one accuracy when the actual tool is in a close proximity to the target, and the other is when the tool is inside in a pre-determined workspace area. To check if the tool-tracking system can fulfil the first requirement, the tool is moved to touch a specific bolt and the system calculates the observed position. The accuracy is the difference between the positions. The timing requirement is tested by calculating the difference in time between when the outgoing tap of the tool is touching the bolt and when the system calculated position of the tool is within the acceptable distance from the bolt.

2.5.2 Test Rig

The test-bench have been designed to simulate industrial conditions, with places with limited line-of-sight. It gives the option to tighten bolts in different orientation. The room where the system is mounted is a simulated industrial environment: The room contains a moving crane that can generate disturbance.

3 Sensor Fusion

3.1 Sensor Fusion Theory

Sensor fusion (or data fusion of sensor data values) is to fuse, or combine, data from different sensors. The combination can be positive, negative or independent, depending on how the measured data change the system uncertainty level [1]. There are many different ways of doing sensor fusion. Akhoundi and Valavi [2] has presented these three categories of sensor fusion:

- Probabilistic methods such as Bayesian analysis of sensor values.
- Least square-based estimation methods and Optimal Theory.
- Intelligent aggregation methods such as Neural Networks and Fuzzy Logic.

Probabilistic methods uses positive combination of sensors. Bayesian filters calculate the conditional probability density of the desired states using Bayes theorem based on measurements from sensors. One of the most well known and most often used algorithms in this category is the Kalman filter [3]. The Kalman filter is a set of equations that estimate the state of the system by iterating through predictions and corrections. The Kalman filter is also an optimal filter because it minimises the error covariance. However, in practice, the systems rarely only have Gaussian probability distribution of the noise, which a Kalman filter assumes, but the filter is still widely used for many applications because of its effectiveness, simplicity and robustness [4].

Another very widely used Bayesian technique for tracking something is a particle filter. The particle filer is a Monte Carlo method that randomly distributes particles in the state space and weighs them according to the probability distribution of the measurement model and the actual measurements to get an estimate. For future iterations, the particles are distributed by the weights calculated and so the particles should converge to the correct state [3]. Particle filtering has the advantage of not needing a Gaussian probability distribution of the noises like a linear Kalman filter does [3], however, it is computationally heavier than a Kalman filter [5]. In terms of computation Kalman filtering is seen as a very efficient method since only values from the previous sample are stored and used for computations [4].

A major weakness of the (original linear) Kalman filter is that it needs a linear model for the behaviour of the system in order to propagate the system and make a prediction for the next sample. However, techniques such as the Extended Kalman filter (EKF) exist which deal with non-linearity by linearising the system. Another development in non-linear Kalman Filtering is the Unscented Kalman filter (UKF) which samples carefully selected sigma points of the state distribution and propagates it through a non linear equation describing the process and uses the propagated sigma points to estimate the new mean and covariance [5]. A distinct advantage of the UKF is that "The UKF calculates the posterior mean and covariance accurately to the 3rd order using(Taylor series expansion) for any non-linearity. The EKF in contrast, only achieves first-order accuracy." [6]. In the least square methods, besides the Kalman filters, there are other methods, such as Optimal Theory, which can be applied on non-linear systems where optimisation theory is applied to minimise or maximise a cost function, often in this case, minimise the mean square error.

Intelligent aggression methods use an algorithm where decision rules, the intelligence, fuses the systems. It can therefore use not only positive dependencies of the data but also independent input from the sensors. In the case of Fuzzy Logic, the sensor data is weighed depending on the characteristics of the sensors [2]. As an example of fuzzy logic, one can look at two sensors, one that gives more correct results closer to a point, and the other that gives the same level of correctness overall. In this case, the data from the different sensors are weighed depending on the position of what is measured. This requires detailed knowledge of the sensors, Neural Networks, however does not. Neural Networks is a technique that weighs the data by a dynamic algorithm, the system in a way, teaches itself how to do the fusion [7].

3.2 Sensor Fusion Implementation

3.2.1 Approach

When deciding on how to do sensor fusion, there are several things to take in account [1]. Firstly the uncertainty should be reduced as much as possible, that is to say that the resolution of the sensors should be considered which, for the systems used and requirements given, was nothing deemed to be an issue. Secondly, the observation interdependence should be considered, if the observations are positively dependant a Bayesian fusion can be used. The systems used are both independent as some look at rotation and others not, and dependant, as more than one system will look at both rotation and position at a time. Other considerations are resource constraints, some of the fusion types demands computational time or a lot of fine adjustments, and robustness, like noise or what happens if a sensor fails. For the resource constraints, the sensor fusion will be implemented to a degree of reliability, however, analysis will not be made extensive. For robustness, no consideration will be taken to if a sensor fails, the system should however deal with the level disturbances stated in the requirements.

For this project there was constraints on time and not a lot of prior knowledge in the area. After careful consideration, and speaking with knowledgeable people in the area (professor Mark T Smith and researcher John-Olof Nilsson, both from KTH), Kalman filtering was chosen as a first priority of fusion to implement. There is a lot of documentation on how to use this kind of sensor fusion and it's also a very common way of solving localisation fusion problems (see 3.1 Sensor Fusion Theory).

The sensor fusion implementation in this project was done with the filtered output from the systems of the individual techniques, and not done directly from the sensor outputs of the different systems. Fusing on the sensor output gives a more accurate measurements since error propagation is avoided. However, this kind of sensor fusion had a much larger dimension and would have taken more resources than available for this project. The developed platform, however, allows the reception of raw data from the systems for filtering, for possible future work.

A detailed plan for the implementation of the sensor fusion was made at mid-term. This plan can be seen in Figure 3. The sensor fusion plan consisted of 5 phases. The first phase being development of a linear Kalman filter, followed by incremental improvements to include rotation and further extend it to an Adaptive Kalman filter in phase 3. Phase 4 and 5 consisted of developing new filters (EKF and Particle filter) to handle non-linearities.

3.2.2 Description of Systems

Firstly, three different coordinate frames need to be defined, the global fixed coordinate frame attached to the test rig, the tool coordinate frame describing the location of the tool point and thirdly the IMU coordinate frame describing the location of the IMU on the tool.

A Kalman filter assumes that outputs from sensors are Gaussian with the mean at the correct value and noise corresponding to the variance σ^2 of the sensor. The variances for all our sensors were calculated by examining the data sheets of the UWB, IMU and SV systems and setting the standard deviation to the reported accuracy of the sensor. The variance was then calculated as the square of the standard deviation σ. This initial approach needed to be modified as it was found that the IMU, UWB system were not performing as specified in the data sheet, and the assumed standard deviation had to be increased.

The output from the individual techniques is a vector for each system are

$$[UWB] = [x_{uwb}, y_{uwb}, z_{uwb}]^T\,, \tag{3.1}$$

$$[IMU] = \left[\ddot{x}_{imu}, \ddot{y}_{imu}, \ddot{z}_{imu}, \dot{\theta}_{x,IMU}, \dot{\theta}_{y,IMU}, \dot{\theta}_{z,IMU}\right]^T \tag{3.2}$$

and

$$[SV] = [x_{sv}, y_{sv}, z_{sv}, \theta_{x,sv}, \theta_{y,sv}, \theta_{z,sv}]^T \tag{3.3}$$

Where x, y, z is the position in the Cartesian coordinate system and θ_x, θ_y, θ_z is the rotation around the three axis. Note that for V_{UWB}, the rotation is not given, and for V_{IMU} the angular velocity and rotation is only given.

3.3 Kalman Filtering

The authors assume some familiarity of the reader with the Kalman filter. Detailed explanations of the Kalman matrices are thus omitted for brevity. [8] and [4] provide excellent introductions to the topic as well as the mathematics behind the filtering.

3.3.1 Line of Sight Filter

For line of sight, LOS, the variables that are of interest in estimating are the positions, are the velocities and accelerations of the tool. Modelling higher order derivatives of the position have the advantage of allowing making better predictions of the state for the next sample. The state matrix X was therefore designed as

$$X = \begin{bmatrix} x & \dot{x} & \ddot{x} & y & \dot{y} & \ddot{y} & z & \dot{z} & \ddot{z} \end{bmatrix}^T, \tag{3.4}$$

where x, y, z are the coordinates of the tool in the global system, $\dot{x}$, $\dot{y}$, $\dot{z}$ are the velocities in the different directions and $\ddot{x}$,$\ddot{y}$, $\ddot{x}$ are the accelerations of the tool.

The measured state matrix Z is a vector of the outputs from the sensor systems and was defined as:

$$Z = \begin{bmatrix} x_{uwb} & x_{sv} & \ddot{x}_{uwb} & y_{uwb} & y_{sv} & \ddot{y}_{uwb} & z_{uwb} & z_{sv} & \ddot{z}_{uwb} \end{bmatrix}^T, \tag{3.5}$$

Assuming all measurements have been transformed to the global frame(See section 3.3.4)
The state transition matrix A, was based on second order equations of motion modelling position, velocity and acceleration:

$$x = x_0 + v\Delta t + \frac{1}{2}a\Delta t^2 \tag{3.6}$$

$$v = v_0 + a\Delta t \tag{3.7}$$

$$A = \begin{bmatrix} 1 & \Delta t & 0.5\Delta t^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & 0.5\Delta t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & 0.5\Delta t^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.8}$$

where Δt is the time between measurements. The measurement matrix M, which maps the measurement Z to the state X is defined as:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.9}$$

It can be seen that the H matrix expresses that there are two position sensors for each co-ordinate corresponding to the UWB and SV system output respectively, and one acceleration sensor for each co-ordinate corresponding to the IMU output.

The state covariance matrix P was initialised as

$$P = c_p \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix} \tag{3.10}$$

where c_p was a chosen initial large value - 500 - corresponding to a large uncertainty in our guessed position during the first iteration of the filter. Since this value is estimated by the filter for further iterations, only initialisation is required.

The measurement noise matrix, which maps how much noise should be expected from each sensor was defined as:

$$R = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{x,IMU}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_y^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{y.IMU}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_z^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{z,IMU}^2 \end{bmatrix} \tag{3.11}$$

where σ_x^2, σ_x^2, σ_x^2 are the variances of each of the sensors. Q is the process noise matrix, which describes how much noise is expected from our process model. For the process noise, since the derivatives of acceleration are not modelled, our process model would exhibit some errors. To simulate this error, it was assumed that the acceleration is constant during each Δt but the sensor output is noisy with white noise corresponding to zero mean. To model this, the process noise matrix is designed as a piecewise white noise (Gaussian distributed with a zero mean) model (see 3.1 Sensor Fusion Theory).

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_a^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_a^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_a^2 \end{bmatrix} \tag{3.12}$$

Where σ_a was set to the expected noise - the Allan variance obtained from the data sheet - in the IMU output during each time step.

3.3.2 Non-Line of Sight Filter

For the non line of sight (NLOS) condition, there is no output from the stereo vision system and thus it is not possible to include it in the filter since it would degrade our estimate. The matrices thus had to be re-sized.

The state matrix was defined as the same matrix as the LOS condition because the same variables are still of interest.

$$X = \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \\ y \\ \dot{y} \\ \ddot{y} \\ z \\ \dot{z} \\ \ddot{z} \end{bmatrix} \tag{3.13}$$

The state matrix was initialized to X_{k-1}, corresponding the last estimate of our system, before the NLOS condition is entered.

$$X = \begin{bmatrix} x_{k-1} \\ \dot{x_{k-1}} \\ \ddot{x_{k-1}} \\ y_{k-1} \\ \dot{y_{k-1}} \\ \ddot{y_{k-1}} \\ z_{k-1} \\ \dot{z_{k-1}} \\ \ddot{z_{k-1}} \end{bmatrix}. \tag{3.14}$$

The sensor output matrix Z had to be changed to include only the UWB and IMU systems:

$$Z = \begin{bmatrix} x_{uwb} \\ \ddot{x} \\ y_{uwb} \\ \ddot{y} \\ z_{uwb} \\ \ddot{z} \end{bmatrix} \tag{3.15}$$

Since there is now one sensor less for each co-ordinate, our measurement matrix also needed to change. The NLOS H matrix was designed to be

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.16}$$

The state covariance matrix P was initialised to the covariance estimate in the previous sample P_{k-1}.

Measurement noise matrix corresponding to our reduced set of sensors was given by:

$$R = \begin{bmatrix} \sigma^2_{uwb_x} & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2 imu_x & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{uwb_y} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{imu_z} & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{uwb_z} & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{imu_z} \end{bmatrix} \tag{3.17}$$

The covariance matrix P and process noise model only depend on the state matrix and thus were not changed for the NLOS condition.

3.3.3 Algorithm

The linear discrete Kalman filter repeats the following steps for each sample time, the notation $\hat{X}_{k|k-1}$, means the estimation of vector X at time k giving the prior observations up to time $k : k-1$.

It starts with the prediction step where firstly the expected position is calculated, the calculations are iterative and to the definition of $\hat{X}_{k|k-1}$, see Equation 3.21, initially this is set at a starting point :

$$\hat{X}_{k|k-1} = A\hat{X}_{k-1|k-1} \tag{3.18}$$

and then the state covariance matrix P is recalculated taking in account the process noise covariance matrix Q :

$$P_{k|k-1} = AP_{k-1|k-1}A^T + Q. \tag{3.19}$$

The second step is to obtain the measurement from the sensors and calculate the Kalman gain K .

$$K_k = P_{k|k-1}H^T(HP_{k|k-1}H^T + R). \tag{3.20}$$

This step of the algorithm corrects the estimation of of the state based on the observation, and the predicted state $\hat{X}_{k|k-1}$ weighed by the Kalman gain:

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k(Z - H\hat{X}_{k|k-1}), \tag{3.21}$$

note that if the actual measurement Z and the measurable predicted state values $H\hat{X}_{k|k-1}$ are the same, the second term will be zero, ergo the predicted state. The new covariance matrix, used in the new iteration is then calculated by:

$$P_{k|k} = P_{k|k-1} - K_kHP_{k|k-1}. \tag{3.22}$$

The steps 3.18 to 3.22 is then repeated for the next sample time.

3.3.4 Rotation Handling

The rotation of the tool was of a great importance in the project. When the tool is outside line of sight it relies heavily on the accelerometer data. Since the accelerometer measures the acceleration of the tool and gravity constantly effects it, the effects of gravity is needed do be removed in our calculations. Rotation was also needed because the point of interest was the tool tip, but both the UWB and the VICON (stereo vision) system measured values that were of other points of the tool than the tool tip. The reason UWB tag was impossible to place at the tool tip and the VICON systems measure point was set to be the same as the position of the IMU so that their rotation axises would align. Quaternions were chosen to be used instead of Euler angles since quaternions can not get Gimbal locked (lose rotational information with some operations).

The conversion from gyroscope data to quaternions is done with eq: 3.23.

$$\begin{aligned} q_0 &= \cos(\theta/2) \\ q_1 &= \sin(\theta/2)a_x \\ q_3 &= \sin(\theta/2)a_y \\ q_3 &= \sin(\theta/2)a_z \end{aligned} \tag{3.23}$$

where a_x, a_y and a_z is the rotation around their respective axis divided by θ and θ can be written as $\sqrt{x^2 + y^2 + z^2}$ according to [9]. The θ is naturally multiplied with the time difference between each iteration to convert it to degrees. This quaternion, which correspond to the angle change between the current iterations is at each iteration added total quaternion by quaternion multiplication (see eq:3.24 [10])

$$\begin{aligned} t_0 &= r_0q_0 - r_1q_1 - r_2q_2 - r_3q_3 \\ t_1 &= r_0q_1 + r_1q_0 - r_2q_3 + r_3q_2 \\ t_2 &= r_0q_2 + r_1q_3 + r_2q_0 - r_3q_1 \\ t_3 &= r_0q_3 - r_1q_2 + r_2q_1 + r_3q_0 \end{aligned} \tag{3.24}$$

where $t = (t_0, t_1, t_2, t_3)$ is the quaternion which corresponds to the total angle change, $q = (q_0, q_1, q_2, q_3)$ is the the latest total quaternion and $r = (r_0, r_1, r_2, r_3)$ is the quaternion which corresponds to the angle change between the last iteration.

The angles from the VICON system is transformed in the same way as the gyroscope angles but there is no need to add quaternions or multiply anything with the time difference between iteration since the VICON system provides with the correct angle straight away.

If a quaternion corresponds to a rotation θ the inverse of a quaternion corresponds to a rotation of $-\theta$. To get the inverse of a quaternion $q = (w, x, y, z)$ eq:3.25 from [11]was used.

$$q^{-1} = \frac{w - x - y - z}{w^2 + x^2 + y^2 + z^2} \tag{3.25}$$

The data from the accelerometer is translated into acceleration in the x, y and z direction. The acceleration vector is then rotated to the real-world coordinate system by using the inverse quaternion of the system. The inverse of the quaternion was used since the accelerometer data is in the IMUs coordinate system and needs to be rotated back to the real world coordinate system. When the values correspond to the real world coordinates the gravity (9.81) is subtracted from the z axis.

All rotations of vectors used eq:3.3.4 [12]. Since the position of the tool tip was the point of interest, and the systems provided a position somewhere in the middle of the tool, there was a need to correct this difference. To achieve this the tool was set in its initial position and then the distance in x, y and z was measured from the sensors measuring point to the tool tip. These vectors were then always rotated to the current rotation and the sensor output was subtracted with the rotated vector to get the tool tip position.

$$v' = \begin{bmatrix} (1-2q_2^2-2q_3^2) & 2(q_1q_2+q_0q_3) & 2(q_1q_3-q_0q_2) \\ 2(q_1q_2-q_0q_3) & (1-2q_1^2-2q_3^2) & 2(q_2q_3+q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3-q_0q_1) & (1-2q_1^2-2q_2^2) \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \quad (3.26)$$

4 System Development

4.1 Individual Techniques

This section describes the individual techniques that are included in the final system design and how they work.

4.1.1 Inertial Measurement Unit

The IMU used is the MPU9060, the other IMUs acquired was not used due to limitations of resources and technical difficulties. The IMU contains accelerometer, gyroscope and magnetometer. The magnetometer values are not used in the system. The IMU is connected to an Arduino Nano which collects the raw data and sends it to the main computer for future processing. The MPU9060 has several different sensitivity settings both for the gyroscope and the accelerometer. The least sensitive setting were chosen for both the accelerometer and gyroscope so that it would handle vibrations during the tightening better.

4.1.2 Ultra Wide Band

The Ultra Wide Band system used in the project is the evaluation kit TREK1000 from DecaWave combined with a EVK1000 evaluation kit from the same company. Both kits combined gives a total 6 units. All the units uses the same circuit boards, EVB1000 (see Figure 6). The EVB1000 can be assigned up to 4 unique anchor addresses and 8 unique tag addresses through physical configuration switches, where anchors are the stationary units and tags mobile [13].

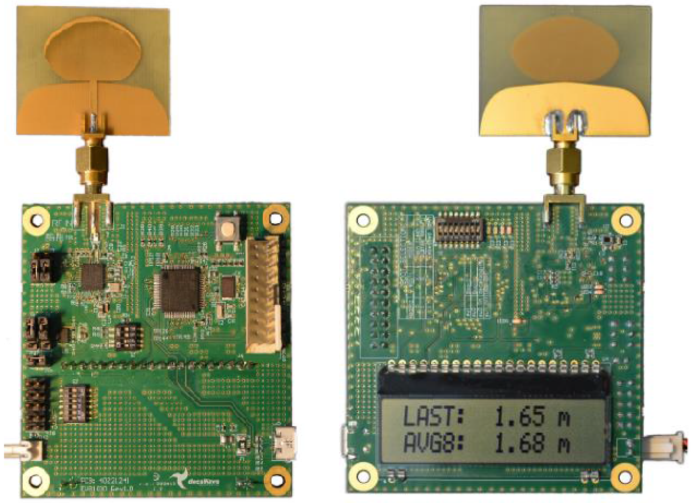


Figure 6: EVB1000, front and back. Picture taken from [14].

The tags operates by sleeping for a period, then waking up and ranging with each anchor in turn (it will stop the sequence if any attempt fail) and then return to sleep [13]. The ranging process consists of the anchor transmitting a signal to the tag and saving the time stamp of the transmission. The tag receives the message and transmits a response back to the anchor after a know delay. The anchor receives the signal and can by comparing the timestamps get the time of flight [15]. After each ranging exchange, the anchor computes the time-of flight estimate and reports the result back to the tag. Anchor 0 is special compared to the other anchors in two regards. It assigns activity slots to all the tags so their ranging doesn't interfere with each other and it reports back the result of all ranging exchanges via its USB port [13]. In this project one tag and four anchors was used. An extra anchor was added since it increases the chance of reliable measurements and to determine whether the tag is above or beneath the anchors [13]. The tag was mounted on the tool while the anchors were mounted on the wall. The software application used in the project, DecaRangeRTLS PC, uses the information through the USB connection with anchor 0. The PC software used in the project is the original software provided by DecaWave, for the TREK1000 application, with some minor alterations. The biggest change is that the current position is broad-casted in a UDP package and received in the sensor fusion program. The default configuration on the EVB1000 boards has a position update frequency of 1 Hz which was assessed as to slow for the intended application. The update frequency was therefore changed to 15,6 Hz.

The TREK1000 kit alone only supports three anchors. With three anchors the system calculates two solutions (one above the anchors and one below the anchors), and chooses the solution below the anchors. This choice is made on the assumption that all anchors are mounted on the same height and are stationed above the area of interest for localisation. A fourth anchor needs to be added to distinguish between the two solutions [13].

Communication in line of sight depends on two factors, the signal level that arrives at the receiver and the sensitivity of the receiver. As long as the received signal is greater or equal to the receiver sensitivity the receiver will be able to detect the signal. The signal strength depends on factors such as transmitted power, losses in the PCB and cables, antenna gains, centre frequency of the channel used and the distance in meters between the transmitter and receiver. Higher frequencies and longer distances causes the signal strength to decrease [16]. There is a phenomenon called multipathing which refers to when there are more paths than the direct path between the transmitter and the receiver. This is generally caused by the radio signal reflecting off a surface. A multipath signal caused by a single reflection will be inverted in phase to the original signal. It is therefore possible that the reflected signal interferes with the direct path signal depending on the difference between the distance they've travelled, buts since the signal pulse is very narrow the bounced signal must reach the receiver within 2 nanoseconds of the direct path signal to cause any interference [16].

Multipath signals have less energy than a line of sight direct path, always travel longer, and therefore arrives later than the direct path. Since it arrives later it is possible to distinguish between direct path signals and signals that has bounced while in line of sight [16].

When the tag leaves line of sight three aspects changes:

- The communication range is reduced since signals overall will be attenuated before they reach the receiver.
- The direct path detection range will be reduced due to attenuation of the direct path signal.
- Time of flight errors will occur due to differences in the refractive index of the material which is obscuring.

These concepts are illustrated in Figure 7.

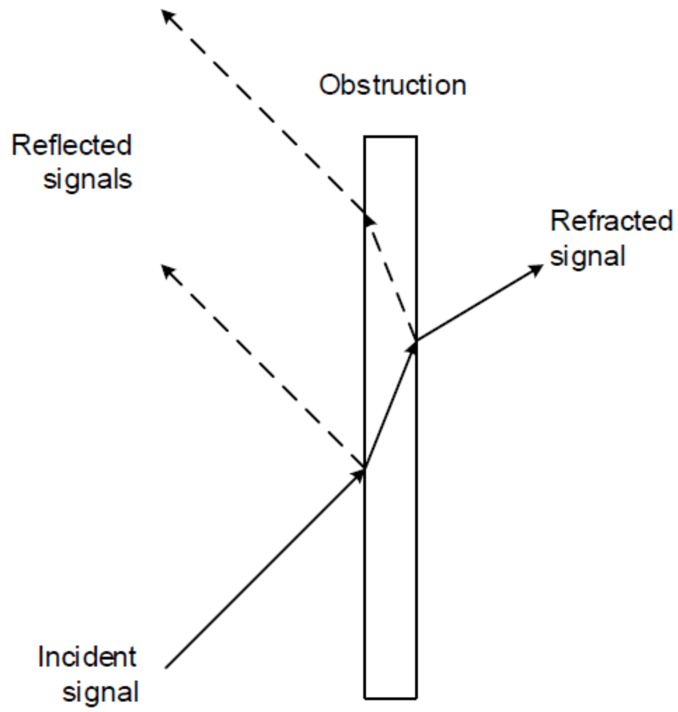


Figure 7: A signal moving through an object, picture is taken from [16].

When a radio frequency signal is obstructed by an object a certain amount of the energy is reflected back from the surface of the object while the remainder goes into the obstruction. A portion of what goes into the obstruction is absorbed by the material (turned to heat).The remainder of the signal is either reflected from the far edge of the obstruction or exists on the other side [16].

Depending on the reflectiveness and thickness of the obstructing object it is possible that the signal strength from the direct path is lower than from the reflected path, or possibly even to low to be detected. If the direct path signal isn't detected then the reported range will be the distance of the reflected path instead. There are however settings that counteracts this behaviour.

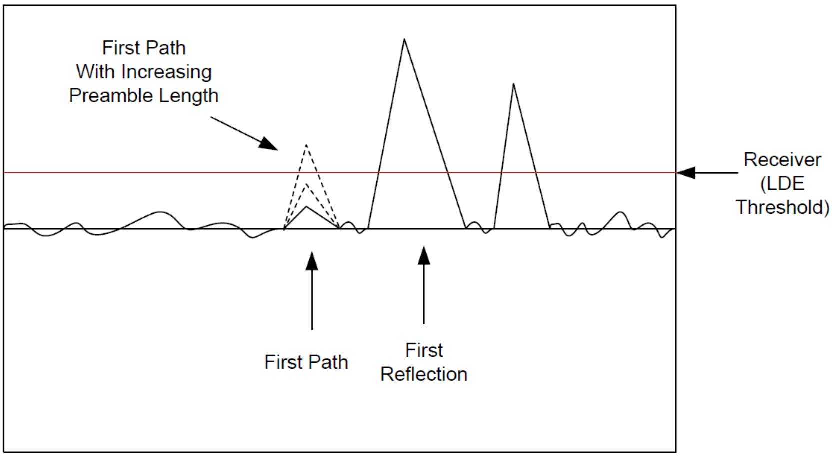


Figure 8: The effects of increased preamble length, picture is taken from [17].

In Figure 8 the first path, or the direct path, with the original settings is below the detection threshold and will therefor be interpreted as noise, and the first reflection will instead be interpreted as the direct path. But if the preamble length is increased then the real direct path signal will be increased. This operation is however only possible thanks to the presence of the first reflection. The presence of the first reflection ensures that the chip receives the preamble symbols and continues to detect these preamble symbols which allows the first path to emerge from the noise floor. If the first reflection is absent then the system would not be able to receive the preamble symbols and therefor the first path will remain below the threshold line [17]. It is however impossible to have the recommended preamble length for non line of sight operations and a high update frequency at the same time.

It is also possible to lower the threshold on the receiver, but if the threshold is set too low the receiver might falsely trigger on noise before the direct path is detected [17]. This was however not used in the final system because of time restraints and complications.

4.1.3 Stereo Vision

The stereo vision system is one of the individual techniques utilised to track the nut-runner. The system used comes from a company called VICON. Six Bonita cameras which work together are calibrated and employed to track the objects created. In addition, VICON software, together with VICON

Datastream SDK, is provided to get access to the data stream of stereo vision system through UDP.

The global coordinate system is set up based on the layout of the test rig. This to obtain the X,Y,Z coordinates more easily and precisely. The coordinate system for the nut-runner is aligned to the global coordinate system when the nut-runner is placed at the zero position holder.

More specifically, as can be seen in Figure 9, the global coordinate system was decided to have the origin at the bolt called origin at surface level. The y-axis and x-axis was aligned to the scales (see the picture below) to be able to use the physical scale on the test rig. In order to get a Cartesian coordinate system, the z-axis thus need to point upwards from the table.

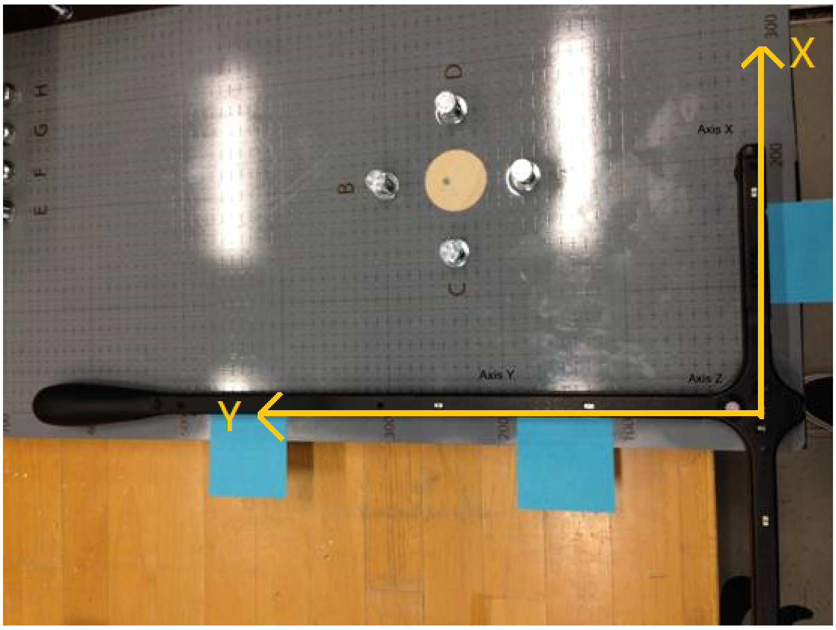


Figure 9: Global Coordinate System

The coordinate system for the tool (a nut-runner) was then decided to align to the global coordinate system when the tool is placed as in the picture below, with the z-axis pointing upwards from the tool tip, the x-axis forward in the nut-runner direction and the y-axis to the left of the tool.

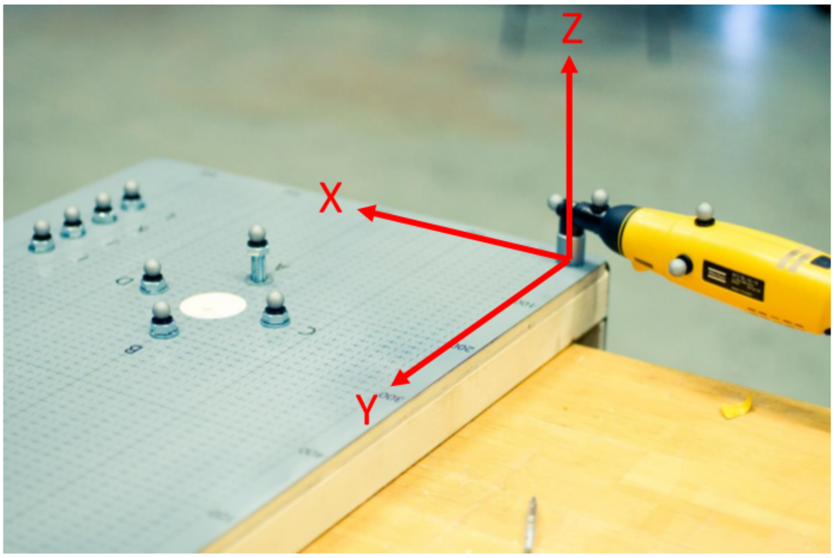


Figure 10: Coordinate System of the Nut-runner

To be able to measure rotation in three dimensions three markers are needed. To increase the reliability of the measurement of the tracked object, two more markers were added. Three of them were placed along the centre line from the tool tip (the position of measure). Then one marker was mounted on each side of the tool in order to increase the chances of getting a good view of at least three markers from as many cameras as possible.

Furthermore, one marker was used in order to place the origin exactly at the tool tip (since it is easier to move an object origin to a specific marker), this marker was then detached in order to be able to tighten the bolts.

Matlab has a prebuilt Simulink file. The code written in Matlab is implemented in order to find which bolt the nut-runner is on. Since the data from the UDP show the coordinates of the nut-runner, these coordinates can be compared with the actual position of the bolts. If the coordinates of the nut-runner are lying within the reasonable range around the position of the bolt, which bolt the nut-runner is on can be shown in the scope. As a result, the global coordinate system created can be verified in Matlab, but the Matlab code was used to verify the test rig together with the VICON system only in the start-up phase.

The stereo vision system sends data via UDP that then is available to collect either in Python or in QT. In the data collection phase for the sensor fusion, the code written in C++ collect data from UDP in the VICON system.

4.2 Final System Design

In this chapter the proof of concept prototype will be described. First through an overview on system level and then of each of the systems separately.

4.2.1 Overall Design

The overall system logic is based on sensor fusion where measurements from the individual techniques are merged together in a high level sensor fusion to support each other during different situations. The logical structure of the system can be seen in Figure 11.

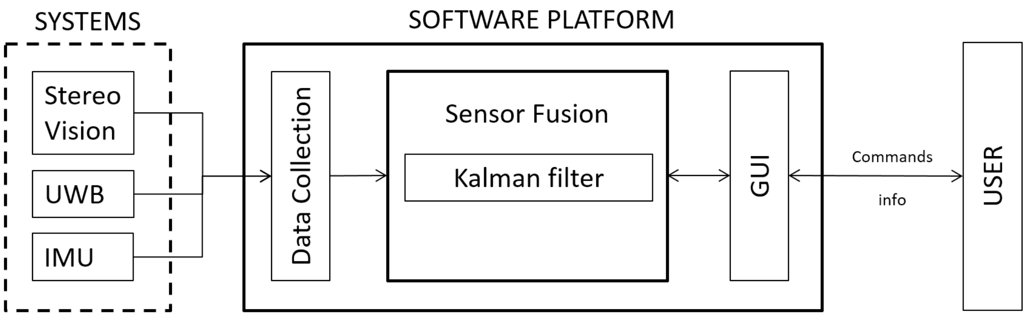


Figure 11: The logical structure of the Final system

To the left in Figure 11 are the individual techniques described in chapter 4.1 Individual Techniques, here called systems. The systems either feed an output position of the tool or as in the case with the IMU, an acceleration to the software platform. The data collection part of the software platform receives the data and store and forward the values to the sensor fusion. When sensor fusion is done the output is visualised via an graphical user interface (GUI) to the user. The system architecture used to achieve the above logical function can be seen in Figure 12.

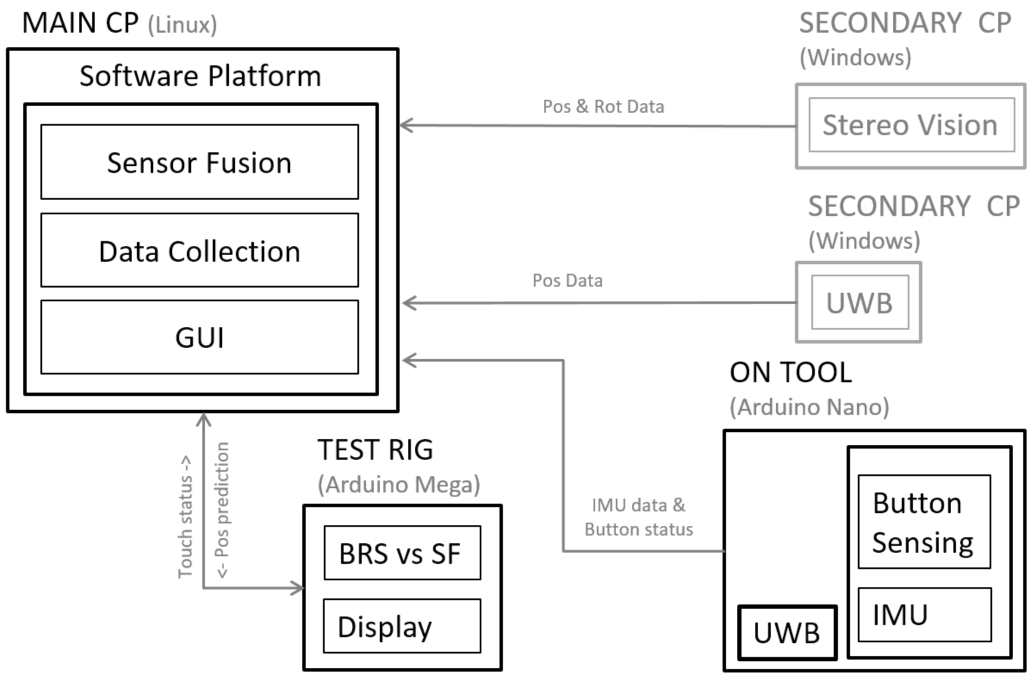


Figure 12: The physical structure of the final system

The architecture is divided into three primary platforms; the main computer, the test rig and the on tool add-on. As secondary platforms, two windows computers have been used in order to facilitate a base for two of the individual techniques; the stereo vision camera system and the Ultra Wide Band (UWB) system. The third individual technique, the IMU is located on add on on the tool. The individual techniques feed data to the main computer where the software platform combine and visualise the result to the user.

From the main computer the sensor fusion output is sent to the test rig in order to visualise the system status to the operator of the tool. The Bolt Recognition System (BRS) on the test rig sense if the tool is touching any of the bolts. The communication between the platforms are either via sending UDP-packages or serial communication, described more in detail under each sub-system. The test rig in turn feed status of the bolts back to the main computer to be able to measure system delay.

In short, there are three main processes in the systems:

- The systems feed data to the main computer

- The test rig send status of the (BRS) to the main computer
- The software platform on the main computer merge the outputs from the individual techniques

4.2.2 On Tool

A fixture have been created to hold the tracking devices on the back of the nut-runner. An Arduino Nano is mounted together with an IMU, an Ultra Wide Band antenna and three stereo vision markers (see Figure 13).

The Nano collects the IMU data and checks if the operator has pushed the trigger to start the nut-runner. The data is then sent over a serial communication to the software platform. The ground on the Nano-board is connected to the tap of the tool, this gives the ability to check if the nut-runner is touching a bolt on the test rig.

The Ultra Wide Band development-board EVB1000 is mounted together with its battery on a fixture located on the battery-pack of the nut-runner.

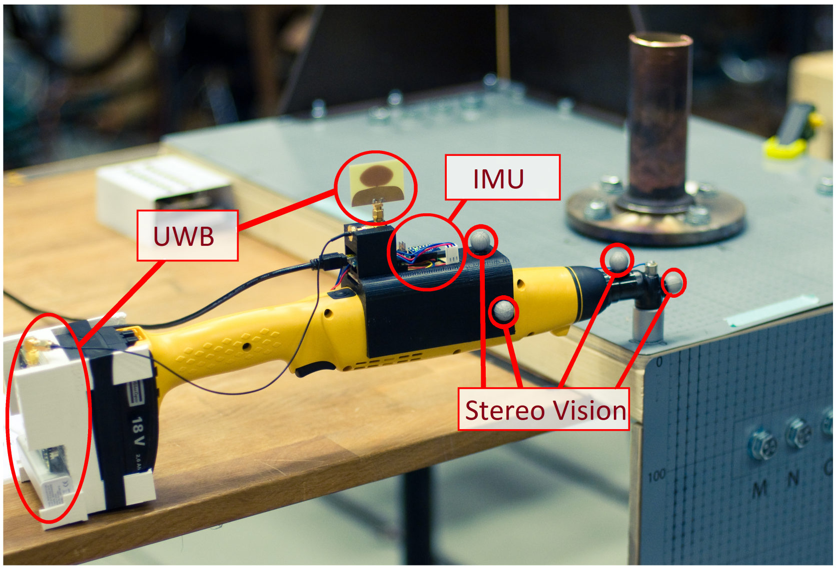


Figure 13: The mounting of the systems on the tool.

4.2.3 Test Rig

The purpose of the test rig is to be able to test the positioning accuracy of the system, and to test the timing constraints of the system as well as to simulate an industrial environment when testing these things.

The test rig consists of bolts in two planes, horizontally and vertically with a sheet metal partially covering the line of sight for some of the bolts (for example bolt H). These bolts are connected via a wire (soldered to the nut on the the backside) to an Arduino Mega that will react if any of the bolts are grounded (see Appendix G). The bolts will be grounded if the tool tip touches the bolt head. This tool tip is grounded via the Arduino Nano on the tool, see Figure 14. This system was named the Bolt Recognition System (BRS). The Arduino Mega controls a LED board (or display) that indicates with the lighting to a corresponding yellow LED when a bolt is grounded, ergo touched by the tool tap (see Appendix I). Eight bolts; A-D, H, I, M and Q; are connected to the Bolt Recognition system.

The Arduino Mega is also connected to the platform via USB and serial communication for receiving when the tool tip touches a bolt and for sending what bolt the platform indicates the tool to be on. The Arduino Mega will then light a corresponding green LED on the LED board. This is to naively indicate that the platform gives the correct results when demonstrating but it is also the measure in which the timing constraints are measured.

In the requirements, it states that the system has to be faster than the average human reaction time of approximately 200 milliseconds. To test for this two functions was created. Firstly the ledboard can give a naive indication; if the green LED (corresponding to the platforms indication) is not appearing to be lit after the yellow LED (corresponding to the touch sensor) by a person, the requirement should be fulfilled. However, this does not give a definitive indication (the measurement is subjective), so secondly, the indication form the test rig is sent to the platform where the platform can compare the time stamp of it's own calculations with the time stamp of the message.

For assuring that the transmission time, and the time it takes for the Arduino Mega to react, is fast enough to give a reasonable value of the delays, a ping-time test was conducted. The ping-time test was conducted as such; a computer would serial send a value to the Arduino, which would then after receiving set one of it's pins high. This pin is then also connected to an interrupt pin on the Arduino, that would, when given a high value, send back to the computer. The computer would record the receiving time and compare it to the time it sent a value. This time would then be used as a worst-case positive delay before the computer could record the time stamp of the received value. This value however, was many magnitudes smaller (it was only dependant on the sampling-time) than the 200 milliseconds delay required that it was discarded.

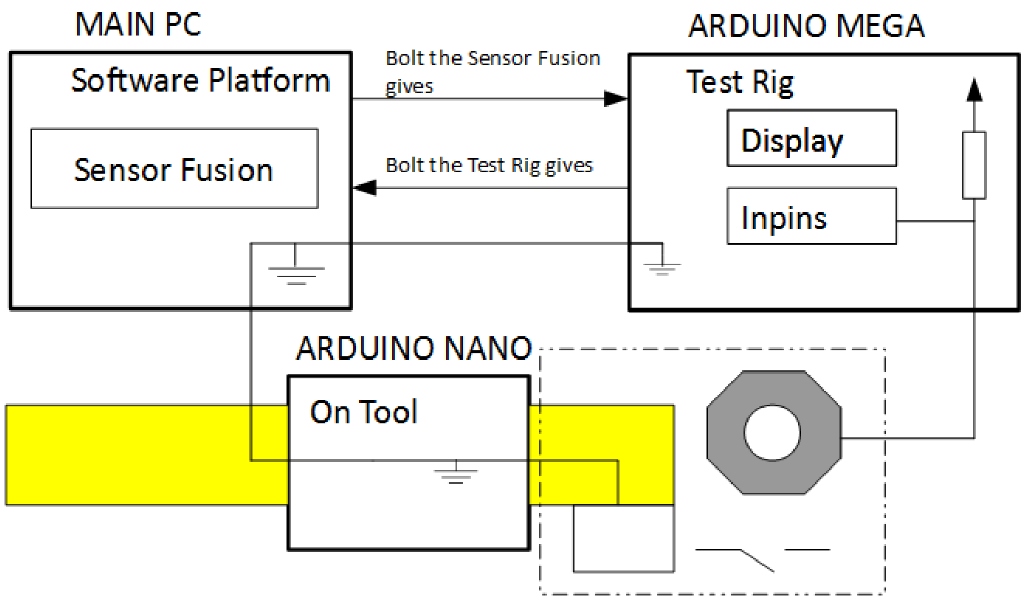


Figure 14: Physical layout of the test rig.

A docking system is built for the nut-runner to have a zero position holder, it was kept as simple as possible. An angle plate, which is attached to the testing table, is employed at origin to hold the nut-runner.

4.2.4 Platform

The platform is on a PC running Linux, it is the central hub which collects all the data from the different systems and stores it so that the sensor fusion code can access it. The code is written is C++, which as a language with low abstraction but still has libraries to facilitate matrix multiplications. To make the coding easier a framework, Qt, was used. The reason behind using Qt framework and not some other framework is because some members in the group had experience of this since before. All data is collected in parallel with each other through different threads.

The stereo vision system and the UWB system are on another PC (running Windows due to the systems limitations) and sends their data over the network via UDP-packages. This was done using Qt framework. The IMU data is sent from the Arduino via serial communication via USB.

All the data is then stored in different queues so that the last thousand values always are accessible and thereafter saved. When a new value comes in the oldest one is discarded. All of it is saved in one class which contains data from all the systems so that every thread and part of the program can access the data. The system calculates the distance from the tool to all the bolts once every 10 ms.

For testing purposes all the data stream are saved individually as separate csv-files. This is done when the user touches a bolt.

The main program also saves the latest calculated position which is accessed by a Python-script. The script then uses a 3D-plotting function to plot where the tool is in 3D-space.

5 Results and System Evaluation

The testing was designed to test the requirements of accuracy and delay as described in subsection 2.5 Testing Method. The accuracy was tested in three stages. First, in a line of sight holding the tool horizontally. Second, holding the tool horizontally in limited line of sight and third, holding the tool vertically in line of sight. The corresponding bolts on the test rig for these cases where bolt A (line of sight horizontally), H (non line of sight horizontally) and Q (line of sight vertically).

The delay was measured by both visually observing if any delay could be noticed and by measuring the time delay between system reaction and physical contact with the bolt. Therefore, on every test cycle and single bolt, the LEDs was observed if any noticeable delay could be observed and the delay was also measured in the system.

In order to gain a reliable test result, many tests cycles need to be conducted. To be able to assume that the test result shows the real accuracy, the null hypothesis that the measurements are wrong needs to be rejected. Although, within the time frame of the project the test sample size was set to 100 in order to be able to reject our null hypothesis with a confidence level of 95%. Hence, a margin of error of 9.8% in the measured values is needed to be accepted.

5.1 Test Data

In order to reach 100 valid test cycles, some extra cycles were conducted. After deleting the faulty test cycles where e.g. a bolt was missed, the result was analysed on the 104 remaining valid test cycles. During each cycle the accuracy was measured for the individual systems, and the accuracy and delay were measured for the system output. The test data from the individual techniques can be seen in Appendix J.1 and the testdata for the final system can be seen in Appendix J.2. The data in these appendices are the accuracy of the different systems which is calculated as the euclidian distance between the system output position and the actual bolt position.

5.1.1 Accuracy Individual Techniques

As seen in Table 1 below, the accuracy for the UWB system is very low. However, it is important to consider the distance between the UWB-antenna and the actual tool tip which could be seen in Figure 13 on page 34. The distance was measured to 21 cm along the x axis, 0 cm along the y axis and -13.5 cm along the z-axis which result in a euclidean distance of 25.0 cm. This distance is affecting the average and median of the accuracy, which means that the easiest way to evaluate the system accuracy is by looking at the standard deviation.

Table 1: Accuracy of the UWB system

Accuracy (m)	A	H	Q
Average (mean):	1.33	1.62	0.911
Standard deviation:	0.847	0.252	0.435
Variance:	0.718	0.064	0.190
Median:	0.994	1.58	0.721

As seen in Table 1, it is between 25 and 85 cm and lowest in the NLOS situation. This is further visualized in Figure 15 as the highly fluctuating errors in distance error. The table and figure are both based on the data from the test cycles and is found in Appendix J.1.

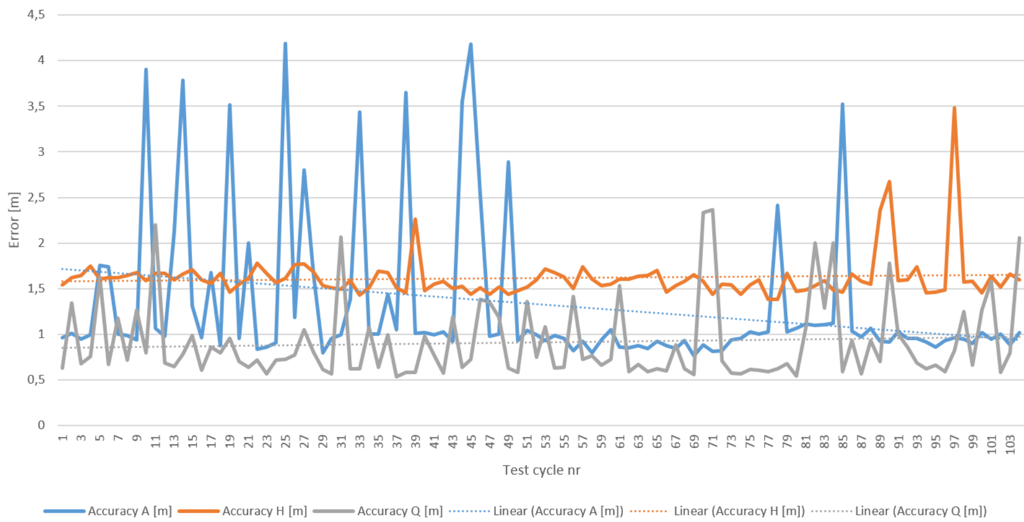


Figure 15: Position error of the UWB system output.

As with the UWB-system, it is important to consider the distance between the point which the SV system measures (which is the position of the IMU) and the actual tool tip. The IMU position on the tool can also be seen in Figure 13 on page 34. The distance was measured to 18 cm along the x axis, 0 cm along the y axis and -9 cm along the z-axis, which result in a euclidean distance from IMU to tool tip of 20,3 cm.

Table 2: Accuracy of the SV system

Accuracy (m)	A	H	Q
Average (mean):	0.214	0.339	0.217
Standard deviation:	0.005	0.085	0.004
Variance:	$2.27*10^{-5}$	0.007	$1.60*10^{-5}$
Median:	0.214	0.382	0.217

As seen in Table 2, the standard deviation for the SV system is 0.4 cm and 0.5 cm in LOS and 8.5 cm in NLOS. The low standard deviation is clearly visible in Figure 16 where the fluctuations (standard deviation) of the SV system is far less than the UWB system seen in Figure 15 above. The table and figure are both based on the data from the test cycles and is found in Appendix J.1.

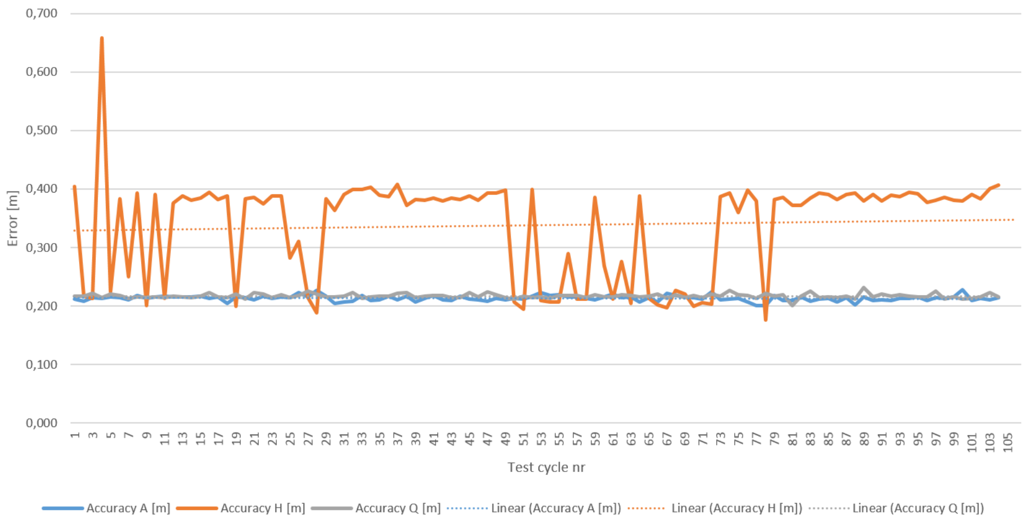


Figure 16: Position error of the SV system output.

5.1.2 Accuracy Final System

The result from the testing of the final system can be found in Appendix J.2. The data is summarised in the Table 3 and visualized in Figure 17 below.

Table 3: Accuracy of the final system

Accuracy (m)	A	H	Q
Average (mean):	0.045	0.491	0.069
Standard deviation:	0.020	0.347	0.018
Variance:	0.00039	0.12009	0.00031
Median:	0.042	0.48	0.064

As seen in Table 3, the fused system has higher accuracy in LOS. This is mostly due to the fact that the stereo vision system is very precise, but nonfunctional without LOS. The standard deviation of the SV system propagates to the final system for bolt A and Q which are in LOS. The standard deviation for the UWB system propagates to the final system in the NLOS situation. Hence, the higher standard deviation for bolt H as seen in Figure 17. The final system brings the behaviour of the SV system into use in LOS situations and have to rely on the UWB and IMU in the NLOS situations.

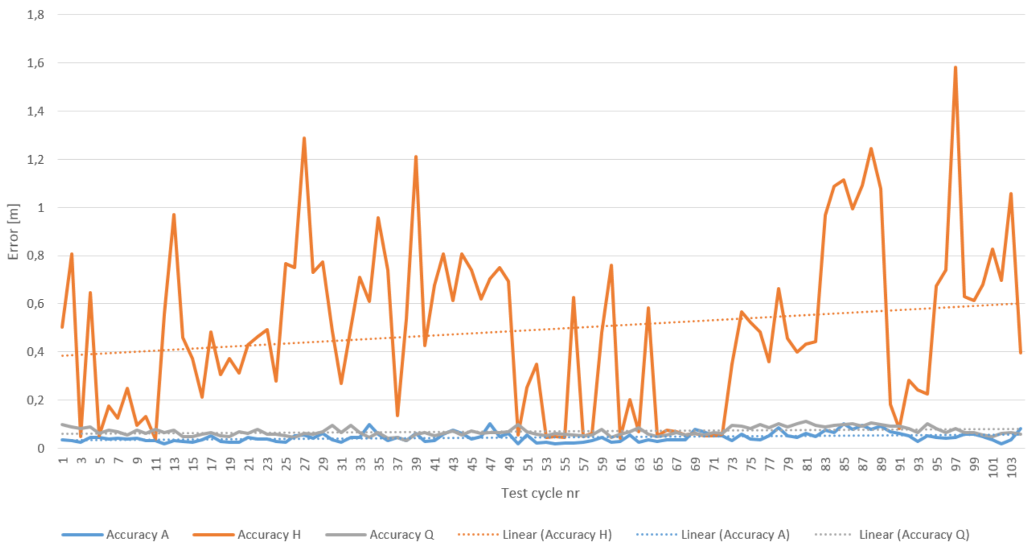


Figure 17: Position error of the final system output.

5.1.3 Delay Final System

During each test cycle, the system delay was tested both visually and measured by the system (as described in chapter 4.2.4 Platform, on page 36). The system delay for each cycle and particular bolt can be seen in Appendix J.2. When the system did not recognise the bolt or the system was after the actual touch of the bolt, this could be seen in the test data as a very high number, usually 30 000 000 ms.

Table 4: Number of cycles with ok delay of the final system

Delay (s)	A	H	Q
observed ok:	80	13	14
observed not ok:	24	91	90
measured ok:	60	10	8
measured not ok:	44	94	96

As seen in Table 4, the system delay was more often regarded as "ok" in the visual observation than by the measurements. This is mostly due to the fact that the measurement in the system only regard situations when the system is before the actual touch of the bolt as ok. In comparison, some delay is regarded as ok when visually inspecting the delay due to human reaction time. In average, when the system recognised the bolt, it was 199.85, 228 and 106,6 ms before the actual touch of the bolt as seen in Table 4.

Table 5: Average time difference of the final system

Bolt	A	H	Q
Average Time difference [ms]	199,85	228	106,6

5.2 Discussion

The system works as an proof of concept prototype for an platform to test and compare sensor fusion methods and different individual techniques. However, it is not able to solve all the requirements at the same time in the current state.

The accuracy requirement of 50 mm is fulfilled only for bolt A and not for bolt H or bolt Q (Bolt Q has an accuracy of 7 cm). Even though the average error is larger than the requirement, the low standard deviation of only 2 cm indicates that the accuracy deviation is due to a static error, e.g. a wrongly made setup or wrongly made code calculations of rotation. The conclusion from the testing is thus that that line of sight situations is not a problem for the system to handle. In fact, using the stereo vision system alone would solve this requirement. Although, the individual systems measure a position of the tool located between the handle and the tool tip. The position therefore needs to be translated to the tool tip position via the rotation of the tool in order to know at which bolt the tool is located. This results in "faulty" measurement values of the accuracy for the individual techniques which seems worse than they actually are. Therefore, when discussing the accuracy of the individual systems, the standard deviation should be considered.

One interesting way to visualise this is by decreasing the average accuracy error of the UWB and SV system by the euclidian distance of the point of measure. This is the same as assuming that system would work ideally in measuring the angle. The values would then change to an average accuracy for the UWB system of 1,08 m, 1,37 m and 0,66 m for bolt A, H and Q respectively. The SV system would have an average accuracy of 1.3 cm, 13.6 cm, and 1.4 cm. This is done in the system by converting the output from the individual systems via the rotation and by deleting the distance vector to give to location of the tool tip. Although, converting it incorporates the error in the angle. In conclusion, the opinion of the authors is that more tests are needed in order be able to do a better system evaluation.

The largest problem for the system is in the non line of sight situations, illustrated in the bolt H measurements. At bolt H, the system needs to rely on the UWB and the IMU, resulting in a decrease in the accuracy significantly. As can be seen in Table 3, the accuracy is less and standard deviation higher in NLOS situations, even higher what the UWB should be able to deliver, one possible cause is that this is due to an improper setup of the UWB system. It is also possible to see that the error at especially bolt H increase over time as seen in the linear trend line for bolt H in Figure 17 which may be an effect of accumulating drift error in the IMU which the system rely on at this bolt.

The accuracy of the system is the same close to the bolts and also within the whole area which the individual systems cover. The system was not specifically tested for the accuracy in a global positioning system and not for the workspace accuracy requirements. However, the system was visually observed when active and it acted the same all over the work space which was defined as the table where the test rig was mounted. Regarding the 2000 mm accuracy requirement in the global positioning system the same could be concluded; the performance does not change while the tool is within the track able area of the individual systems.

The system output is the position of the outgoing tap of the tool in Cartesian coordinates in three-dimensional space and the rotation around the three cartesian axes. The normal vector from the outgoing tap could be calculated from these values, although, it is not done at the moment and the orientation of the tool was not tested. The origin of the system is a fixed point in the workspace on the test rig as shown in Figure 10 on page 31.

The requirement on the system delay was set in order to avoid "human noticed delay" of the system position output. The human reaction time was defined as 200 milliseconds and this was both observed visually by identifying LED order and by conducting measurements in the system. As with the accuracy, the system performed far better on bolt A where the observed ok and measured ok result were 80 and 60 out of 104 respectively. In the situations when the delay was not ok, it was mostly due to the fact that it did not find the bolt at all. When the system recognised the bolt, it was in average 199.85, 228 and 106,6 ms before the actual touch of the bolt A, H and Q correspondingly.

The robustness of the system is well enough to be able to deal with partial and, in some way, non line of sight. The IMU and the UWB system gave a reasonable output, even if the location was not as precise as the requirements state. However, it is the belief that this accuracy could be acquired. No further disturbances was simulated than the normal activity of the lab-hall the tests were conducted in. More disturbances can be found on in a factory, specifically for the ultrasound. These kind of disturbances (sound, magnetism, light, vibrations) need to be investigated and tested with since filters probably will have to be made to deal with them. The system as it is now is not ready to be taken out into production, the cord and the bulkiness of the tool will not make it robust enough. It can also be manipulated by the operator if the operator covers the stereo vision markers, this is something that can't be avoided. Because of this, the motivation for using more than one system is even larger. The UWB would in this case still give an output.

The update frequency is at least 10 Hz as required by Atlas Copco. For the ultrasound, this could have been a problem, so for all systems added or exchanged, this needs to be taken into consideration.

The system takes use of cameras for the stereo vision, however, they will only track the objects specified, therefore it will limit the intrusiveness because the identity of the operator will be concealed. However, to feel that something is intrusive is a subjective experience. The authors of this report make no claim on completely fulfilling this requirement, but have to the extent possible made sure to mitigate the risk of this experience for operator.

The price of the stereo vision system, VICON, is well above the budget set for the whole project. Access to this system was however given by the Mechatronics Department at KTH, and therefore it could be used. Because of the modular design of the platform, this was not seen as a problem since this system should be exchangeable.

5.3 Future Development

The solution presented in this report has not been optimised and refined for the best sensor-fusion result but is, rather than a solution, a platform to develop sensor fusion for indoor localisation (or tracking) on. It is also intended as a proof of concept for using a modular design and sensor fusion to solve this problem.

5.3.1 Individual Techniques

The individual systems used in the sensor fusion conducted in this report might be over- or under sized for the problem. The stereo camera system is well over budget (see Appendix F), however, this system can be more economically justifiable if used for a larger area, as you can add cameras to the system. This system also fulfils the requirements of accuracy alone i line of sight. It does not, however, output the orientation in quaternions, which will have to be implemented (see 3.3.4 Rotation Handling).

In order to fully evaluate this modular way of doing sensor fusion more care should be set on how to filter the individual techniques. The IMU should to be filtered with greater care, and the gyro should be reset by the other systems to mitigate drift.

In order to further develop on this platform, the systems used can be changed to further test for the optimal sensor fusion for a specific area. The future of the individual techniques can also be investigated, there are developments done in this area, and the price of the systems might drop significantly, and the accuracy might increase. For the solution presented in this report, the individual technique of ultrasound was judged to be a good alternative to stereo vision, so this alternative can be evaluated further. The choice of specific IMU shall also be thoroughly considered, since it's accuracy will greatly affect the non line of sight accuracy.

5.3.2 Sensor Fusion

The sensor fusion algorithm could be developed further to deal with non-linearities in the process model. This can be achieved by means of using a:

- Adaptive Linear Kalman filter
- Non-linear Kalman filtering
- Monte Carlo methods - eg. Particle filters.

It is the authors opinion that the adaptive Kalman filter is an extremely simple way of dealing with non-linearity and should be considered before investing resources into non linear filtering. The main principle is increasing the process noise on observing that the error in system prediction for the next sample is above a certain threshold. The authors have attached an example implementation of this in appendix I.3.

For non linear Kalman filtering, the Extended Kalman filter (EKF) or Unscented Kalman filter(UKF) could be considered. The EKF is essentially the linear Kalman filter but using first order Taylor expansions of the non linear equations. The UKF takes a more monte carlo approach to deal with the non linearity by sampling selected points from the prior and posterior distributions of the state to calculate the mean and covariance.

The particle filter is another very effective algorithm for tracking applications and should also be considered if the computational complexity is acceptable.

5.3.3 Platform

The physical layout of the solution is now on several computers which in line with the desired modularity, however, for keeping down cost and communication times, different sensor systems might be allocated into one computer. For the end product of a localisation system, this will be needed to be implemented.

The modularity can also be developed further, right now the platform has functions for translating in-data into the format it uses for doing sensor fusion, if access was given to the individual techniques systems, this could be done in their end and the platform would be easier to set up.

The system is also able to a further extent save data, this does not improve the sensor fusion for a Kalman filter, but might for other types of filtering.

The press of the button on the tool could also be an input to the platform, this could help with handling disturbances from the motor on the IMU and other possible systems.

The platform also has a user interface that if optimised, could aid the sensor fusion developer to give a better overview if it could display standard deviations and results in real time or in report format after a test run.

5.3.4 On Tool

The microprocessor on the tool is now connected to the computers with a wire, for this to have practical use, the communication will be switched to wireless communication, where delays in the transmission will have to be considered for the sensor fusion. The size of the systems are now much larger for a usable system, but all systems could be re-sized within the development of the final localisation system. The system could also be made more robust, both physically and electrically.

5.3.5 Possibilities of a Localisation System

To be able to determine where a tool is located opens up for many new possible applications within the tool. There is a possibility of having the tool logic communicating with the sensor fusion platform, and having an automated tightening regime for the different bolts depending on their location within the workspace. This could decrease the the human error. The authors see it possible to make sure that the system is robust enough not to create a stop in the production.

6 Conclusion

As can be seen in the chapter 5 Results and System Evaluation, the system fulfils requirement of an accuracy of 50 mm for bolt A but not for bolt H or Q. However, when going into NLOS situations and only relying on the UWB and IMU the accuracy decrease significantly. The low standard deviation of the accuracy indicates a statical error and it might be due to an improper setup of the UWB system, or a coding error in the sensor fusion platform. The system works rather well in LOS situations when the accuracy of the SV system can be used. When the bolt was recognised, it was most of the times made before the actual touch of the bolt.

In conclusion, the indoor localisation or tracking of a tool could possibly be done with this method. The individual techniques used in this solution, and the implemented sensor fusion, show a proof of concept of this possibility. The different systems work together to overcome individual obstacles and weaknesses in each system. However, the measured data only give an indication of due to the relatively few testing cycles and testing difficulties. The authors believe that with more time it might be possible to fulfil all the requirements in full and that there is a value to further investigate and test these kind of solutions.

References

[1] A. C. S. Chung and H. C. Shen, "Entropy-based markov chains for multisensor fusion," *Journal of Intelligent and Robotic Systems*, no. 29, p. 161–189, 2000.

[2] M. A. A. Akhoundi and E. Valavi, "Multi-sensor fuzzy data fusion using sensors with different characteristics," *arXiv preprint arXiv:1010.6096*, 2010.

[3] P. Hafner, T. Moder, M. Wieser, and T. Bernoulli, "Evaluation of smartphone-based indoor positioning using different bayes filters," in *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, Oct 2013, pp. 1–10.

[4] G. Welch and G. Bishop, "An introduction to the kalman filter. 2006," *University of North Carolina: Chapel Hill, North Carolina, US*, 2006.

[5] H. Wang, "Improved extend kalman particle filter based on markov chain monte carlo for nonlinear state estimation," in *Uncertainty Reasoning and Knowledge Engineering (URKE), 2012 2nd International Conference on*, Aug 2012, pp. 281–285.

[6] E. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, 2000, pp. 153–158.

[7] H. Takeuchi, K. Yamauchi, and N. Ishii, "Self-supervised learning and recognition by integrating information from several sensors," in *Industrial Electronics Society, 2000. IECON 2000. 26th Annual Confjerence of the IEEE*, vol. 2, 2000, pp. 1195–1200 vol.2.

[8] S. Thrun, *Probabalistic robotics.* MIT Press, 2005.

[9] A. developers, "Motion sensors," 2016-01-06. [Online]. Available: http://developer.android.com/guide/topics/sensors/sensors_motion.html

[10] Mathworks, "Quaternion multiplication," 2016-01-06. [Online]. Available: http://se.mathworks.com/help/aeroblks/quaternionmultiplication.html

[11] E. W. Weisstein, "Quaternion." 2016-01-06, from MathWorld–A Wolfram Web Resource. [Online]. Available: http://mathworld.wolfram.com/Quaternion.html

[12] Mathworks, "Quatrotate," 2016-01-06. [Online]. Available: http://se.mathworks.com/help/aerotbx/ug/quatrotate.html?requestedDomain=www.mathworks.com

[13] *Moving from trek1000 to a product*, decaWave, 2015, version 1.0, Avaiable: http://www.decawave.com/support.

[14] decaWave, "Evk1000 presentation," 2014. [Online]. Available: http://www.decawave.com

[15] *The implementation of two-way ranging with the DW1000*, decaWave, 2014, version 1.0, Avaiable: http://www.decawave.com/support.

[16] *Channel effects on communication range and time stamp accuracy in DW1000 based systems*, decaWave, 2014, version 1.02, Avaiable: http://www.decawave.com/support.

[17] *Non line of sight operation and optimization to improve performance in DW1000 based systems*, decaWave, 2014, version 1.30, Avaiable: http://www.decawave.com/support.

Appendix

A Pre-Study

Tooltracker
Pre-Study

Report in Mechatronics HK
MF2058
MAY 30, 2015

Written by

Stefan Annell	steann@kth.se
Satyajit Bagchi	sbagchi@kth.se
Adam Ekström	adameks@kth.se
Christoffer Lindahl	clinda@kth.se
Tobias Lundin	tolundi@kth.se
Lin Xuan Lyu	lxlyu@kth.se
Björn Magnusson	bjomag@kth.se
Mattias Nilsson	mattiani@kth.se
Elin Nordmark	emkno@kth.se
Michael Refors	mrefors@kth.se

The Royal Institute of Technology
2015 KTH
Stockholm

Abstract

This State of the art report investigates the possibility to create an indoor positioning system for an industrial environment with an accuracy of 10 mm. The research project is conducted as a joint project together with the external client Atlas Copco. The research method has been a deep-scan of current state of the art solutions and techniques for indoor positioning systems. They have been evaluated against the requirements set together with Atlas Copco. Furthermore, the evaluation included the authors' prior knowledge about the techniques and confidence in the techniques. The result shows that there are techniques possible to solve the requirements one by one. Although, there is no solution capable of meeting all of them. The next step of this project has two options; continue with the same requirements at a higher risk or alternate some of the requirements to decrease the risk of the project outcome. Finally, a family of four concepts have been developed.

Contents

Abbreviations

AC	Alternating Current
AoA	Angle of arrival
DC	Direct Current
DOF	Degrees of Freedom
CoO	Cell of Origin
EPAM	Extended Phase Accordance Method
IMU	Inertial Measurement Unit
IR	Infra Red
LED	Light Emitting Diode
MIT	Massachusetts Institute of Technology
NASA	National Aeronautics and Space Administration
NLOS	Non Line of Sight
LOS	Line of Sight
RF	Radio-frequency
RFID	Radio-frequency Identification
RMSE	Root Mean Square Error
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
SLAM	Simultaneous Localisation and Mapping
SPL	Sound Pressure Level
TDoA	Time Difference of Arrival
ToF	Time of Flight
UWB	Ultra-wideband

1 Introduction

1.1 Background and Objective

The objective of this report is to give a pre-study to solve the problem of: *Localising a tool inside a manufacturing plant in order to make automatic settings to the tool depending on location.*

In the industry, the need for knowing how much to tighten a bolt is a part of quality assurance. Today these settings are done by hand for handheld tools, automatic settings would eliminate human error.

1.2 Requirements

Requirements are a vital part in a product design process. Requirements are a major guideline for defining the possible technologies, or what direction the state of the art research needs to be heading.

1.2.1 Functional Requirements

Accuracy:

- The system shall be able to measure the central position of the outgoing tap of the tool with accuracy of 10 mm when at a distance of 600 mm or closer to the target.
- The system shall be able to measure the central position of the outgoing tap of the tool with an accuracy of 500 mm inside a work space.

Output:

- The system shall update the position of the tool with a frequency of at least 10 Hz.
- The system shall output the position of the tool in the form of a Cartesian coordinate system in three-dimensional space (X, Y, Z) and the normal vector of the outgoing tap.
- The origin of the Cartesian coordinate system shall be defined as a fixed point in the work space.
- The systems position lag shall not be greater average human reaction time, estimated to 200 milliseconds.

1.2.2 Soft requirements

Accuracy: The system shall be able to measure the central position of the outgoing tap of the tool with an accuracy of 2000 mm when located in a global indoor positioning system.

Intrusiveness: The worker should not feel supervised and/or tracked.

Robustness:

- The system should be robust enough to not create a stop in the production.
- The system should be robust enough to operate in an industrial environment.
- The system should not be manipulable by the operator.

Cost: The cost of the tracking system is important however it is not a constraint for the state of the art.

Scalability: The system should be easy to install/remove/change, however it is not a constraint for the state of the art.

Dimension: A possible add-on to the tool should not affect the usage significantly. The size and weight of the add-on should preferably be of similar dimensions as the current add-ons.

1.3 Scope

In this report a pre-study to above specified requirements is presented. The report includes state of the art, both for unmitigated techniques and for techniques that require some sort of sensor fusion or combination. The study only looks at the techniques that are for indoor positioning and localisation.

1.4 Method

The techniques are evaluated using an advanced decision matrix. This technique is introduced by Pugh.[1] In order to evaluate the techniques, the requirements are weighed against how hard or important they are, the sum of the weighing factors is one. Thereafter the techniques are scored a number between zero and one by the authors of this report. The scoring is done on two parameters. Firstly, confidence on the system satisfying the requirements and secondly, knowledge of the system. The scoring by the authors and the weighing factors are then factored and the average of the scoring decides the final scoring for the technique.

2 State of the Art

In this section the current state of the art in indoor positioning will be presented. Both basic technologies that could be used to solve the problem and complete commercialised, solutions will be presented.

2.1 Vision

There are three identified techniques for positioning with vision. They are Camera on Tool, Stereo Vision and Time of Flight cameras. This section includes these three techniques and one more, Structured Light, which is a combination of cameras and projected light patterns.

2.1.1 Camera on Tool

Visual odometry (estimation of change position over time) is the process of estimating the position and orientation of an agent (e.g., vehicle, human or robot) using only the input of one single or multiple cameras attached to it. [2] The relative position error of visual odometry with a single camera is about 1.2 percent on a 20 m path. Stereo visual odometry provides more accurate trajectory estimates, with relative position error ranging from 0.1 to 2 percent. [2] The accuracy is up to 2 cm, which is given by David Nister. [3] Small vision sensors are quite cheap, less than 50 USD for single pieces. The possibility to implement this technique in a rough environment is low. Rapid movements and obscured camera views are two big challenges.(Mårten Björkman, pers. comm.) The robustness issues should be taken into consideration. This technique is used on NASA Mars Exploration Rovers. [4]

2.1.2 Stereo Vision

Stereo vision is a method to compute a 3D-Object from two or more images. The images must be of the same object, from different viewpoints. [5] The distance between the cameras and the object can be calculated by observing the pixels that corresponds to the same point in the 3D-object. The point can be calculated in a 3D space, by knowing the focal length of the cameras, baseline and view angle. [6]

There are mainly two possible usages of stereo vision. These are stereo vision cameras mounted on the tool or cameras mounted in the environment. Stereo vision cameras mounted on the tool is not applicable for this particular problem since the size of add-ons on the tool is limited. Furthermore the stereo cameras need a baseline that will be too long to function properly and motion blur will be a problem. It's hard to implement this in a rough environment.

Cameras mounted in the environment is promising. More than one camera can cover a bigger area and can follow one or more tools at the same time. Stereo vision could be used in the environment as a global indoor positioning system to give a rough position with around 2000 mm accuracy. With this technique, we can also get a more accurate zone positioning less than 5 cm. It updates at more than 10Hz, which fulfil our requirements. [7] The cost depends on the amount and quality of the cameras, but normally it is not expensive. The focal length and the actual distance to the target determine the accuracy which can be achieved less than 5 cm. Also, it can fulfil the implementation in a rough environment. The drawbacks of the technique are that line of sight is needed and that the target may be blocked by the operator. However, this technique, with other techniques combined, can be very useful in the project.

2.1.3 Time of Flight

Time of flight uses a method where it sends out a laser pulse, and measures the time it takes for the pulse to bounce and come back. By knowing the speed of light, the distance can be calculated. A 3D ToF camera works by illuminating the scene with a modulated light source, and observing the reflected light. The phase shift between the illumination and the reflection is measured and translated to a distance. [8]

The company Mesa Imaging uses the time of flight camera for positioning. [9] A fully packaged industrial product, the Mesa 4000 and 4500 costs about 4500 USD, although, it could be built from scratch to a cheaper price. The resolution of this technique is about ± 2 cm at 0.5-5 m, with a measuring distance of 0-9 m. [9] The product is feasible to implement in a rough environment. However, after talking with Patrik Jensfeldt(Patrik Jensfeldt, pers. comm.), this technique does not seem very promising, since there will be problems such as motion blur, pointing the camera in a relevant directions and blocking of camera. Besides this, if the camera is aimed at very smooth surfaces, like metal, the light will bounce straight off, since it will not scatter.

2.1.4 Structured light

Structured light is the process of projecting a known pattern onto a surface, then by using a vision system record the displayed light on the surface. By observing the distortions, the depth and surface information can be calculated. From this a 3D-model can be created which in turn is possible to use to calculate a distance . [10] Structured light is often used as active triangulation, because it has one structured light-source and one camera. Infrared light, laser or a normal video projector can be used as structured light. To use structured light mounted on the tool and then record the target area can be a technique that could solve the problem. [11]

The cost of this technique starts at 500 SEK, which is very cheap. The best accuracy of this technique is less than 0.01 mm. [12] The possibility to implement in rough environment is medium. A problem will be that if the tool moves a lot, the system will be influenced and become inaccurate.

2.2 Light

This section contains short information about different positioning techniques which utilises some kind of light.

2.2.1 Laser - Time of flight

Time of flight lasers works by sending out a laser pulse, which reflects against an object in the environment and the time of the arrival of the return pulse is converted into a distance. There are systems which can give an accuracy of 9 μm [13], but they are expensive. Time of flight is also used in laser measurement units in construction and it is possible to get a laser measurement with a better accuracy than 1 cm for less than 500 SEK. [14] Laser measurement units are used in construction sites and are robust enough for an industrial environment. The technique has no latency of measurements and can be run at a higher frequency than 10 Hz. The downside with the technique is that it is only possible to measure in one dimension, requires a very good temporal accuracy, requires line of sight and can easily be manipulated by covering the sensor.

2.2.2 Laser - Triangulation

Triangulation with lasers can be done by sending a laser beam towards an object. When the reflected beam returns it is captured by a lens which focuses it and causes it to hit a very precise point of a sensor. The distance depends on where the laser hits the sensor. With this technique accuracies down to 0.04 μm can be achieved depending on the work range used. Typically distances between 2 - 1250 mm are measured. [15] The measurements does not have any delay and can be updated more frequently than 10 Hz. The laser triangulation systems are robust, extremely accurate and built to work in industrial environments. The downside with the system is that it is expensive and, gives only a one point measurement and it can easily be manipulated by covering the sensor.

2.2.3 Visual light communication

Visual light communication utilises image sensors and planted light sources to triangulate the position of an object. The light sources are used as reference points for the the optical receiver. Each LED has its own ID which is represented by a binary code which is broadcasted by switching the light on and off with high frequencies. There are several different ways to triangulate the position but the one which have achieved the highest accuracy is triangulation by signal strength. [16] Another benefit with RSS (Received Signal Strength) is that there is no need for synchronisation between the transmitters (light sources), which also makes it easier and cheaper to deploy the system.

The LED:s are inexpensive and a regular off the shelf LED can handle the switching frequency required. The photo sensor can be bought relatively cheap as well (a common mobile camera is enough). The best physically measured accuracy achieved is 0.651 cm (RMS of the error). [17] This accuracy was measured in 2D and not 3D (height was known). It is possible to measure in 3D but simulations in other articles indicate that the noise tolerance decreases and the mean of the positioning error increases [18] to almost the double in 3D.

The technique is very interesting since it offers higher bandwidth capacity than radio based solutions. It is also robust to electromagnetic interference, has an excellent security and does not cost much to install. [19] The problems with the technique is that it suffers from shadows and reflection. There are also problems with the angle of the photo sensor and that the system is very easy to manipulate. The system might also be of interest for less accurate position estimations. [19] There is at least one commercial system which utilises the technique in combination with mobile camera, IMU and bluetooth signal strength to track people in a store. [20]

2.2.4 Projected IR light

The Optotrak Certus system is able to have an accuracy of 0.1 mm at a range of 7 m. [21] [22] When combined with active markers, which is basically IR light sources, it can track 6 DOF wireless when in line of sight. This is a high end research level system and it is quite expensive.

The well known Kinect system from Microsoft is based on IR light technology where there is one IR emitter that sends out light and one IR Depth sensor which captures the depth of a 1280 x 960 grid which creates a point cloud. Even though this sensor is considered cheap, it is quite potent in accuracy. [23] The problem compared to the other Time of Flight cameras is that it is not designed to work in a industrial environment. Khoshelham and Elberink [24] measures and evaluates the resolution and accuracy of the Kinect, and it was concluded that it should not be used at a longer range of 3m, since it will have too big random measurement errors and too long distance between the points. The accuracy is roughly between 0.1 cm at 50 cm to 1.5 cm at 300 cm.

2.3 Sound

Sound is a mechanical wave that travels through air or other media which theoretically makes it possible to use in a non-line of sight (NLOS) situation. Techniques used today most commonly use sound waves in the spectrum of ultrasonic sound, although, solutions using audible sound also exists.Techniques using audible sound are not feasible in our situation since it would be uncomfortable for the operator.

Techniques or algorithms used for sound positioning are Time of Flight (ToF), Time difference of arrival (TDoA), Angle of arrival (AoA) often referred to as Triangulation, Extended Phase Accordance Method (EPAM) or Sonar.

Ultrasonic solutions create a cheap alternative to other positioning systems, it is capable of generating high accuracy and, as mentioned earlier, capable to work in a NLoS situation. The downsides of the techniques are first and foremost the decay profile of sound, reflections in environment and sensitivity to background noise. Depreciation due to the spherical diffusion of sound waves makes this technique limited to a distance of 10 m. [25] In an industrial environment there are plentiful of sources generating ultrasonic noise and specifically in environments using air tools where noise levels can be as high as 100 dB [26], decreasing the range of use further. Other downsides are the sensitivity to change in temperature (this is often solved by using temperature sensors on transceiver and receiver, although temperature between them are not possible to measure) and a rather long response time, at worst 10 Hz due to the speed of sound. [25] One last aspect to consider is the working environment for the user of the tool we need to position. There are several guidelines regarding exposure levels, e.g. "...the Canadian 1991 [...] recommendations give 110 dB SPL as the maximum level for occupational exposure in a 1/3 octave band centered on 40 kHz". [27] Ultrasound affects the user and especially if the transceiver is placed at the tool (which means close to the user) the sound levels are restricted.

2.3.1 Angle of Arrival

One of the most interesting solutions for this project are the AMT Ultrasonic add-ons. These are basically a product able to position a tool in an industrial environment. In the datasheet for their product they claim to have "exact positioning" and by judging the branding; down to mm accuracy. [28] They use triangulation (Angle of Arrival) to position the tool. This is a product built for same situations as this Tooltracker project and use almost identical conditions as defined in our requirements (with exception to technical details about accuracy, which is not possible get to know from the datasheet).

2.3.2 Time Difference of Arrival

Wendeberg et al. (2011) [29] conducted a project using the TDoA method and they were able to reach an RMSE accuracy of 2.5 cm by using their own non-linear "Cone Alignment" algorithm. In order to use this technique they needed to be able to synchronize the receivers up to an order of 0.1 ms. They have tested their system in real world situations.

2.3.3 Time of Flight

The MIT cricket system [30] is a system solution where beacons are used to send ID and data. This is made trough ultrasound and radio to get an ID and time-stamp, making the receiver (the dynamic target) able to make a distance calculation to every transceiver. This is a very scalable system with sub 10 cm accuracy at 8 m and about 1 cm at 2 m (depending on the angle). The orientation accuracy was under 3 degrees between -60 to 60 degrees for the angle between the perpendicular to the receiver array and a beacon.

The Active Bat prototype [31] from 1997 uses a similar technique as the Cricket system. They claim to have an accuracy of 3 cm and do this by setting up a grid of ultrasonic receivers in the ceiling with a distance of 1.2 m. They then measure the ToF from the transceiver. Harter (2001) [32] uses a similar solution which by improvements of new transducers reaches resolutions under 1 cm. These systems in combination with the cricket system might form a promising solution.

A variation of this is tested in the Teliamade system [33], where an ultrasonic transmission is used to calculate time of flight and a zigbee module is used to communication information between the nodes. The next step after that system is to use MEMS based sensors that are intrinsically omnidirectional and have greater range.

One last company working with the ToF method is Hexamite. [34] By using a ToF calculation in arriving signals and by looking at the doppler effect it is possible to calculate both position and velocity of the transceiver. They claim to have a precision of the system of 2 mm in their HX19V3 system. [35] This is an interesting ready-made product that is possible to buy and test for validity in our case. The price of a system is around 1500-2000 euros depending on the components.

Liu [36] predicted in 2007 that one future trend is the combined solutions for indoor positioning with ultrasound. How to combine them into a practical system is a topic of sensor fusion. In Table 1 some companies using sound for positioning are presented. A more extensive table from 2012 could be seen in Indoor positioning technologies [7] at page 56 and from 2007 in a survey of wireless indoor positioning techniques and systems [36] at page 1077.

Company name	Year	Reported Accuracy	Principle	Application
AMT UltraSonic	2013	"exact positioning"	AoA	Product
Wenderberg et al	2011	2.5 cm	TDoA	Prototype
Cricket	2005	1-2 cm	ToF	Product
Teliamade	2013	9.6 mm	ToF	Research project
Hexamite	2011	0.2 cm	ToF	Product

Table 1: Companies using sound

2.4 Radio

Radio-based positioning techniques, such as those employed by cell phones, WiFi, and Zigbee are some of the most commonly used positioning techniques due to their relatively high transmission power and the requirements of dedicated devices. The positioning methods used in radio-based techniques can be broadly listed as:

1. Cell of Origin(CoO)
2. Angle of Arrival(AoA)
3. Time of Arrival (ToA) or Time Difference of Arrival (TDoA)
4. Received Signal Strength(RSS) based trilateration
5. Location fingerprinting

A typical RFID system has three components: a transponder or tag (located on the object to be tracked); an interrogator or reader (which receives the information from tags); and a control unit (which operates the system and processes the information). The basic idea of the RFID technique is to transfer the information of identification from an electronic data-carrying device - the RFID tag - to an RFID reader via a RF interface. The technique can be classified into two systems - passive and active - depending upon the signals transmitted and the tag structures. A passive RFID tag contains very simple components to respond with its information of identification to the signals triggered from a RFID reader. It does not contain an electronic power source itself. The energy for the RFID tag's circuit is transmitted from the RFID reader via magnetic or electromagnetic fields over a short range (less than 3 m). Its reading range is limited by the range of energy transition. In contrast, active RFID tags have a longer reading range (over 15 m) due to built-in batteries. In positioning applications, RSS is the major observation component in RFID systems. It is used to determine either the appearance of a mobile user in the reading range or the distance between transmitters and receivers. [37] In Table 2 below a summary of companies using radiowaves for positioning is presented.

2.4.1 Ultra-wideband

Ultra-wideband (UWB) is a radio technology for short range, high bandwidth communication holding the properties of strong multipath resistance. To some extent also used for its penetrability of building material which can

Company name	Year	Reported Accuracy	Type	Principle	Application
Utchitomi	2010	20 cm	Passive	CoO + AoA	Simulation
Fujimoto	2011	15 cm	Passive	CoO + Range	Study
NaviFloor	2011	50 cm	Passive	CoO	Product
Baum	2007	100 cm	Passive	CoO	Development

Table 2: Companies using Radio

be favourable for indoor distance estimation, localization and tracking. A typical UWB setup features a stimulus radio wave generator and receivers which capture the propagated and scattered waves. In contrast to narrowband operation, UWB waves occupy a large frequency bandwidth (>500 MHz).

The low power spectral density prevents harmfulness to the human body and bounds the interference of UWB signals with other narrowband receivers. The low frequency components in the UWB signal spectrum have the ability to penetrate building material such as concrete, glass and wood [38]. This is a useful property for indoor positioning, because it enables ranging under NLoS conditions and makes inter-room ranging possible. On the other hand, partial signal penetration into the target object is unfavourable for precise distant measurements, because the reflected signal includes multiple returns besides the outer boundary reflection. Therefore robust extraction of useful information from the received signal is a major challenge in UWB ranging. Superposition of different scattering effects complicates the data interpretation. [7] In Table 3 below some companies using UWB for indoor positioning are presented.

Company name	Year	Reported Accuracy	Type	Principle	Application
Ubisense	2011	<15 cm	Active	TDoA, AoA	Product
Zebra	2011	<30 cm	Active	TDoA	Product
TimeDomain	2012	2 cm	Active	ToF	Product
decaWave	2010	<10 cm	Active	TDoA	Product

Table 3: Companies using UWB. Note that the accuracy listed is in LoS.

2.4.2 Wi-Fi trilateration

A transmitter send out a Wi-Fi signal and a receiver collects the signal and based on the signal strength determines the distance to the transmitter. A distance between the transmitter and receiver can be calculated by comparing a RSSI value and a function curve that describes signal strength versus distance. By having multiple transmitters in an area, the position of the receiver's position can be predicted. In Table 4 some examples of existing system using Wi-Fi trilateration are presented.

Company name	Year	Reported Accuracy	Type	Principle	Application
Navizon	2015	3 cm/m	Active	RSSI	Product
Ekahau	2015	3 cm/m	Active	RSSI	Product

Table 4: Companies using Wi-Fi trilateration.

2.4.3 Wi-Fi fingerprinting

A radio map is created by measuring the RSSI data from several transmitters in an area. Probability distributions of RSSI Values is generated by the data, that corresponds to a given positions in the area. By measuring RSSI and comparing it to the fingerprints, a position can be predicted for the receiver. In Table 5 some examples of existing system using Wi-Fi Fingerprinting are presented.

Company name	Year	Reported Accuracy	Type	Principle	Application
Indoors	2015	$<$5 cm/m	Active	RSSI, Fingerprint	Product
WiFiSLAM	2013	2.5 cm/m	Active	RSSI	Product

Table 5: Companies using Wi-Fi Fingerprinting.

2.4.4 Bluetooth trilateration

RSSI signals from Bluetooth beacons with unique signature. The beacons positions are known locations in an area. The receiver uses triangulation to calculate its position.

2.5 Inertial Measurement Unit

An inertial measurement unit, IMU, is a gyroscope in combination with an accelerometer. [39] [40] The accelerometer is able to measure the acceleration in the three axes of the IMU body frame, and the gyroscope measures the angular rate of rotation in three axes. [41] Best accuracy possible is time dependent because of the quadratic growth in position error due to double integration of acceleration data containing bias and drift errors, [42] [43] [44] but is very accurate in short periods of time and the accelerometer can be reset and the error can be bound in moments of zero velocity. There are also techniques to find out the sources of error for an accelerometer: bias, drift, non-linearity etc. It can also be integrated with other methods for binding the error further. The best accuracy feasible with smaller gyroscopes is 0.3 deg/root-hour with 0.1-1 accelerometer bias. [45] This information can be used to estimate and bind the error. The estimated price of this technique is 100-5000 SEK.

2.6 Magnetic

Magnetic localisation techniques uses artificially made magnetic fields. It could either be permanent magnets, AC fields or DC field and the localisation is achieved by having several sources of magnetic fields and then using a triangulation technique for finding the position and the attitude of the source as described in. [46] The problems this technique faces is that when ferromagnetic materials is close to the sensor, the sensor picks up on the material which disturbs the measurements. There has been some work done to map and reduce this noise at Stanford University. [47]

2.6.1 Artificial AC magnetic field

The system called G4 [48] is a wireless localisation system using AC magnetic fields, the system is developed by Polhemus. It has a reported accuracy when in LoS of a few milimeters in a 6 DOF with a working area of 1.5 m, which in turn could be added on to about 5 m depending on the amount static sources that is used. This technique has the advantage of working in NLoS situations, with some loss of accuracy, but the loss of accuracy is still below 5-10 cm depending on what materials that obstruct the view (if it is not a ferromagnetic material).

2.7 Hybrid Systems

2.7.1 Sensor fusion with UWB, IMU and a dead reckoning system

In Sillero's masters thesis [49] a number of sensor fusion algorithms using Kalman Filters and Particle Filters are developed to improve the localisation accuracy for a mobile robot. Based on simulations it was concluded that the sensor fusion algorithms significantly improved the localising performance over the individual systems, and accuracies around 4-5 cm was achieved. However when trying the algorithms in a real environment with a Ubisense system and an IMU the performance did not improve significantly over the Ubisense system itself, which had an accuracy of about 40 cm. The conclusion from that experiment was that the IMU used was too inaccurate to provide any improvements to the Ubisense system. This study still shows that using sensor fusion might be a good approach to solve the problem.

2.7.2 Systems based on sensor fusion without beacon instrumentation

Bhuiyan et al. [50] conducted a study about indoor positioning for a mobile robot without beacon instrumentation. Sensor fusion algorithms with Kalman Filters using WiFi signal strength and IMU:s as sensors was developed. The accuracy that the sensor fusion algorithms achieved was about 2.9 m with a standard deviation of 2 m.

A commercial system with a similar approach is a system called NEON, developed by TRX Systems, for indoor 3D positioning. NEON tracks persons in GPS-denied areas by fusing measurements from different sensors like IMU, pressure, light and Time of Flight RF ranging. The system is based on TRX:s SLAM algorithm FeatureSLAM and operates without beacon instrumentation. The typical accuracy for NEON is 3-7 m. [51]

These systems are included in the State of the Art to show what kind of accuracy that is reasonable to get with a completely mobile system without any infrastructure installed.

2.7.3 CLIPS

Tilch and Mautz invented a system called CLIPS [52], which stands for Camera and Laser based Indoor Positioning System. CLIPS utilises fusion of lasers and cameras to determine the position. It uses one camera and a device that project several distributed laser spots. This is used as a reference frame. With this they can reduce the computational costs, because they only use one camera. The system is also mobile since the device creates its own reference frame. The standard deviation in one axis, for one measurement, was about 0.2 mm. During what distance this measurement was made is unstated in the article. But they claim to have a deviation of about 2 mm/m.

2.7.4 FARO

ABB uses a system called FARO laser tracker which utilises a Time of Flight laser. The system returns the 3D position of the object with a measured resolution of 0.009 mm at distances up to 3 m. The system uses the distance measurement combined with with the angle of the laser to achieve this. FARO can locate its target with the aid of cameras and uses a sphere-mounted reflector to bounce the laser back. The downside with this technique is that it requires line of sight, has a limited angling span of the sphere mounted reflector and is extremely expensive. [13]

2.8 Summary and Comparison

A summary of the techniques can be seen in A Appendix. You can see an overview of how the different techniques fulfil the requirements set in the introduction. It can be noted that the unit measurement differ between the techniques, this is kept in this way because the implicit increasing or subtraction of accuracy this can give if changed. The same can be said for the currency displayed, the exchange rate and the difference market (and shipping) might give a false value if changed to only one currency.

3 Discussion

3.1 Advanced Decision Matrix Evaluation

In Summary and Comparison of the state of the art techniques (A Appendix) you can see an overview of how the different techniques fulfil the requirements set in the beginning of the report. In order to evaluate this summary of the noteworthy techniques, firstly the requirements have been weighed against how hard or important the requirement is (see 1.4 Method), results are presented in Table 6 and the indivudiall scoring can be seen in (C Appendix).

Requirement	Scoring
Accuracy of 10 mm when at a distance of 600 mm or less from targets	23.75
The system should be robust enough to be able to operate in an industrial environment and the system should be robust enough to avoid causing stop in the production	15
6 degree of freedom output	13.125
Accuracy of 500 mm inside a work space and accuracy of 2000 mm when in a global indoor positioning system	9
Position update with a frequency of minimum 10 Hz	8.625
The system should not be able to be manipulated by the operator	7.75
The systems position lag shall not be greater than 200 milliseconds	7.625
Worker should not feel supervised and /or tracked	5.25
The size	4.875
and weight	3.625
The cost of a tracking system	1.375

Table 6: Requirement scoring, higher scoring means higher importance of the requirement.

Thereafter the different techniques have been scored by the authors of this report (see 1.4 Method). The results of this can bee seen in Figure 1 and Figure 2. Techniques that clearly aren't suitable for the requirements are left out of this comparison. As well as IMU as it is time-dependent and also regarded by the authors of this report as a complementary technique to deal with non line of sight (seperate section below). The promising techniques according to the grading given by the authors of this report (see Figure 2) are Ultra-wideband, time of flight sound, stereo vision, triangulation and angle of arrival of sound, time difference of arrival sound, time of flight laser, FARO, and UWB, IMU dead reckoning. Where only Ultra-wideband, stereo vision, FARO, and UWB, IMU dead reckoning are able to give a position with six degrees of freedom. Stereo vision can give an accuracy below 5 cm, which means that it may be a good candidate to solve the problem, maybe in combination with another more accurate (but maybe one dimensional output) technique. The sound techniques has shown to have at least a medium robustness in industrial environment, which might seem like a problem initially, however, these techniques still will not get down to the accuracy given in the requirements as will not the hybrid system UWB, IMU and dead reckoning.

However, the ranking of these systems might be dragged down by the authors knowledge, in Figure 1, it is also clear that the confidence in the magnetic field and the Projected IR light is high, which both have mm precision and can measure 6 degrees of freedom.

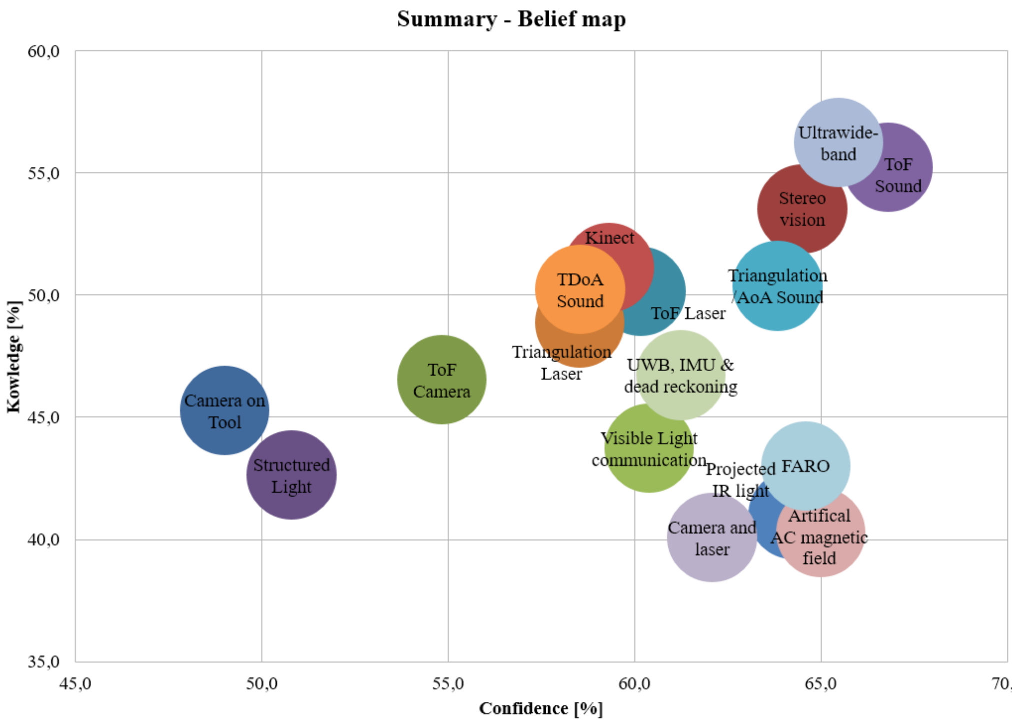


Figure 1: Summary of the authors' confidence and knowledge regarding the different techniques

	Satisfaction Score [%]
Ultrawideband	38,8
ToF Ultra Sound	38,0
Sterio vision	36,0
Triangulation and AoA Ultra	34,1
Time of flight laser	31,9
TDoA Ultra Sound	31,6
Kinect	31,5
FARO	30,9
Triangulation Laser	29,8
Projected IR light	29,2
Visible Light com	28,8
Artifical AC magnetic field	28,7
UWB, IMU & dead reckoning	28,4
TOF Camera	26,8
Camera and laser	26,1
Structured light	23,4
Camera on Tool	23,0

Figure 2: Summary of the authors' satisfaction regarding the different techniques

In the summary it can be seen that some of the systems does not fulfil the requirement of intrusiveness, however, this is a subjective experience and therefore, if these systems show a lot of potential, they should be evaluated further with a focus group (or similar). The individuall scoring of each technique can be seen in (B Appendix).

3.2 Non Line of Sight

In the requirements it can be seen that line of sight is not evaluated as a criteria. However, that the system loses line of sight is a very possible situation, therefore systems that fulfil this criteria is interesting to consider. The systems that still work without line of sight is IMU, Ultra-wideband, magnetic systems like G4, time of flight camera and camera on tool. The IMU have the advantage of working without any fixed references but the gyro will drift. Usually the gyro is reset, but in order to do that the tool has to be still, which can be problematic. The gyro might instead be gradually corrected which can be done by different methods and help from other techniques. Camera on tool and time of flight camera may have problems when it is directed towards something it does not recognise, for this to work best it should be moved slowly and steadily. How it would work on a handheld tool is something that needs to be evaluated, but as can be seen in the scoring, camera on tool has the lowest satisfaction score and time of flight camera has a lower scoring as well. The magnetic solution does not have a very high satisfaction score, it scores very low on robustness, as in an industrial environment it would be a lot of disturbances from the tools in the factory. The Ultra-wideband, however, scores the highest of all the systems, however it does not fulfil the accuracy (best is 2 cm).

3.3 Accuracy Comparison

Accuracy comparison of the investigated techniques can be seen in Figure 3 below where the requirement of 10 mm is marked by a red circle. The most accurate of these techniques is FARO and laser - time of flight, shortly followed by vision - structured light. However FARO is very expensive with an satisfaction score of 31.7 (see Figure 2), which is medium. Time of flight laser only has one dimensional output but is ranked as a top choice and the structured light might become inaccurate depending on how the operator moves the tool and is therefore ranked very low. Most of the techniques do not work without line of sight but is very advantageous in other aspects and combined with a non line of sight technique can create a system that will work in many situations.

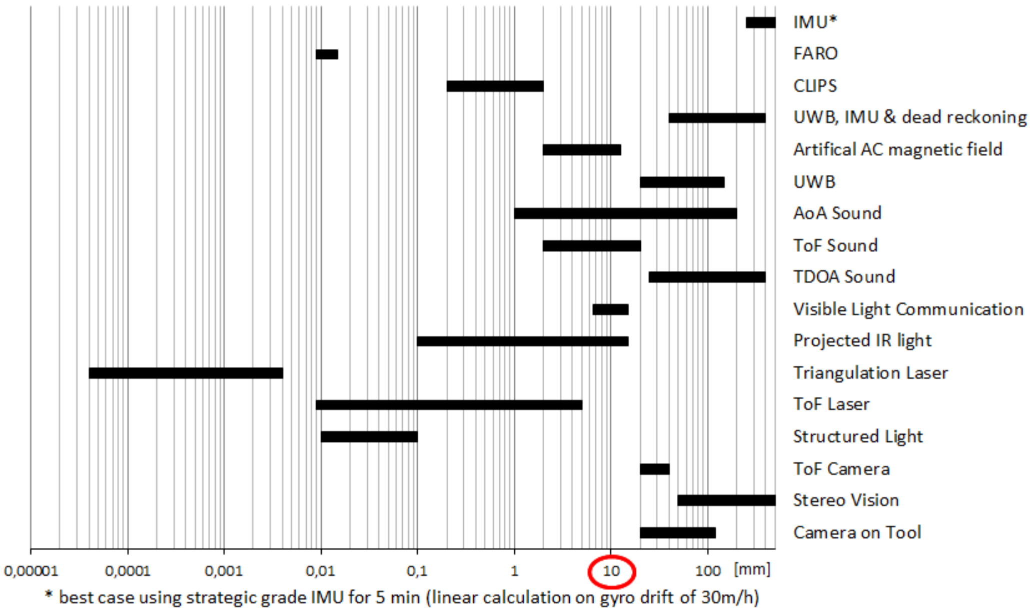


Figure 3: (Logarithmic) Accuracies for different techniques [mm]
* best case using strategic grade IMU for 5 min

3.4 Conclusions from the State of the Art Evaluation

The conclusion that can be given is that if a system shall be developed with the kind of precision that are stated in the requirement, there will be a higher risk of project failure. However, there is a clear possibility of being able to build a system with about 5 cm accuracy.

And as can be seen by the evaluation, building for a different requirement of accuracy is an entire set of techniques. However, the goal can perhaps be reached by easing another requirement. To change the accuracy requirement might make this pre-study invalid.

4 Proposed solutions

In this section a general system design that could solve the localisation problem and four idea concepts that incorporates different technologies will be presented. The concepts were developed at a brainstorming session.

4.1 System design

The proposed system design is based on fusion of different sensors that are suitable for different categories of localisation. By using sensor fusion with e.g. a Kalman Filter a better estimate of the system state can be achieved than from using the sensors individually. The categories of localisation considered are:

- Global positioning
- Local positioning
- Non-line of sight positioning

Global positioning would provide a rough estimate of where the tool are within the factory with a sub-meter accuracy. Ideally this would work both with and without line of sight. The most suitable technology for this is Ultra-wideband, but vision and sound could also be used. Local positioning would provide an accurate position with less than 1 cm error within the work station. For this, technologies such as external vision, lasers and sound could be used. Because of the environment where the tool will be used, non-line of sight positioning is also needed. For NLOS positioning IMU:s and cameras mounted on the tool could be used.

4.2 Concepts

Here the four brief concepts generated at a brainstorming session will be presented. The concept are preliminary and may not all be possible to build in reality or may be very hard to implement. The concepts will not be evaluated in this report since they were generated before the scoring of the different technologies was performed. They are presented only to give an estimate of what a prototype could look like.

4.2.1 Concept 1

The first concept uses UWB for global positioning, an external vision system that detects two light stripes on the tool for accurate local positioning and an IMU for local positioning, see Figure 4. By detecting the light stripes with more than one camera and by knowing the geometry of the stripes the pose of the tool could easily be calculated. The light could be non-visible to the human eye to not cause any distractions to the workers and the external vision system could filter all other wavelengths so the workers wouldn't feel supervised.

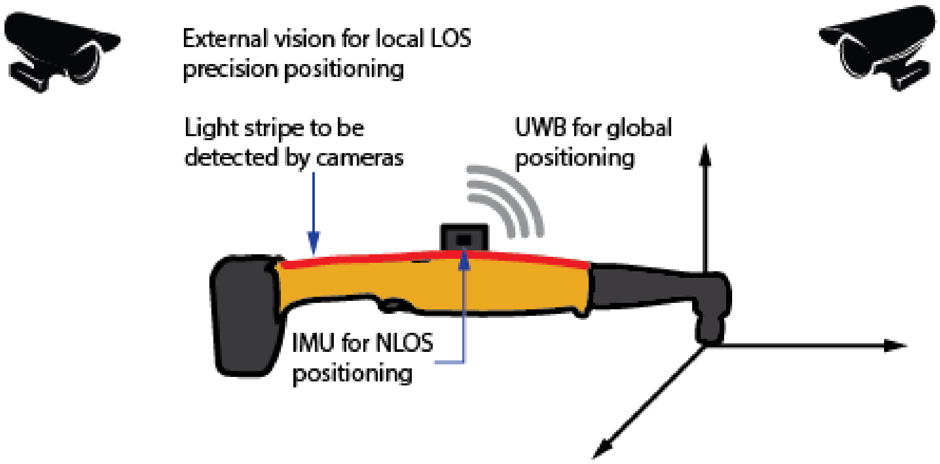


Figure 4: Concept 1

4.2.2 Concept 2

The second concept, see Figure 5, uses sound localisation for accurate local line of sight positioning. This concept assumes that a map of the environment is known and that the IMU can be calibrated at every bolt to reduce drift. The IMU will handle all non-line of sight positioning. It assumes that sound localisation and IMU provides sufficiently good accuracy for global positioning when fused.

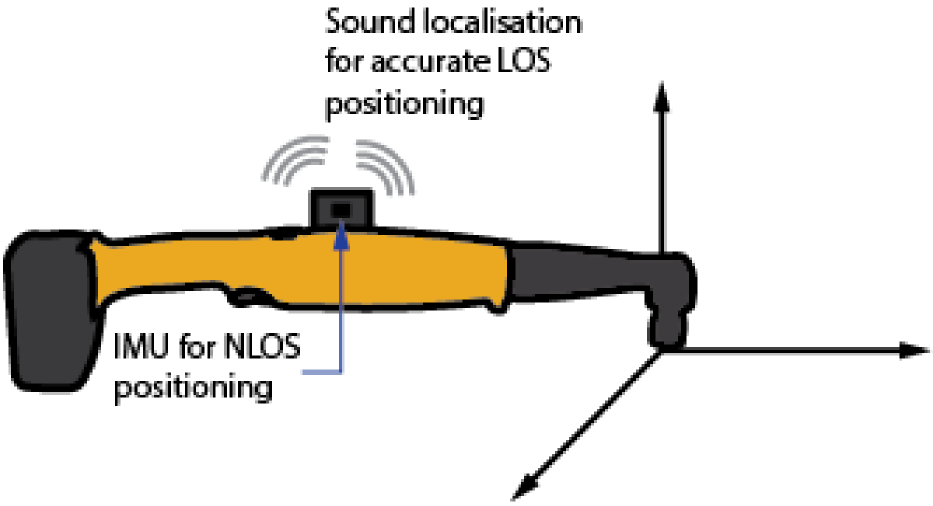


Figure 5: Concept 2

4.2.3 Concept 3

The third concept, see Figure 6, uses UWB for global positioning and an IMU for non-line of sight positioning. It uses a Time of Flight camera that matches its output to a map for local positioning. The ToF camera would also work without line of sight, which improves local NLOS positioning.

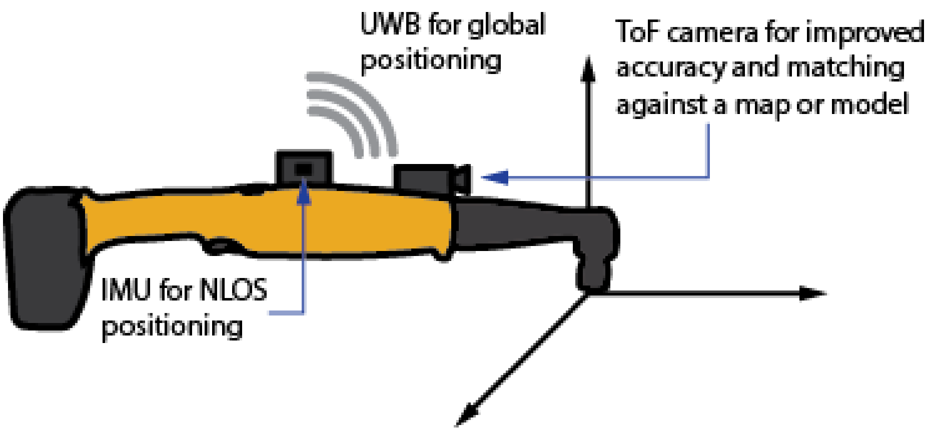


Figure 6: Concept 3

4.2.4 Concept 4

The fourth concept, see Figure 7, uses an external vision system for global and local positioning and an IMU for NLOS positioning. The time of flight laser is used to generate a very accurate position when it detects the tool.

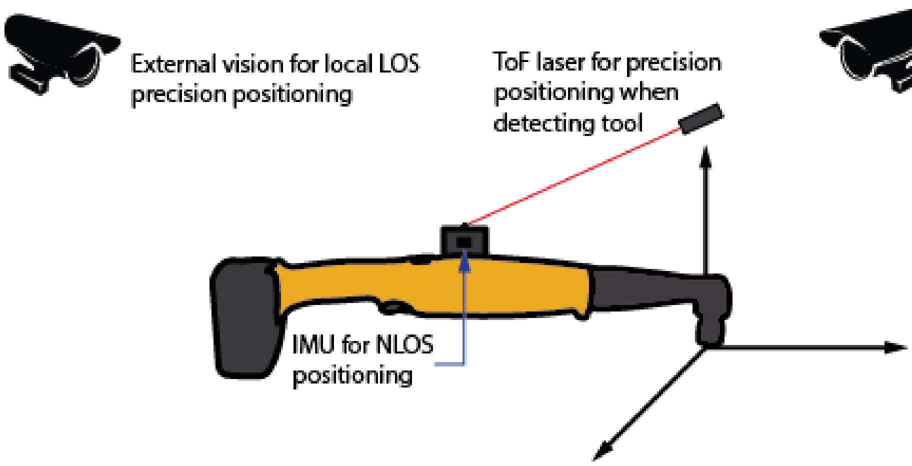


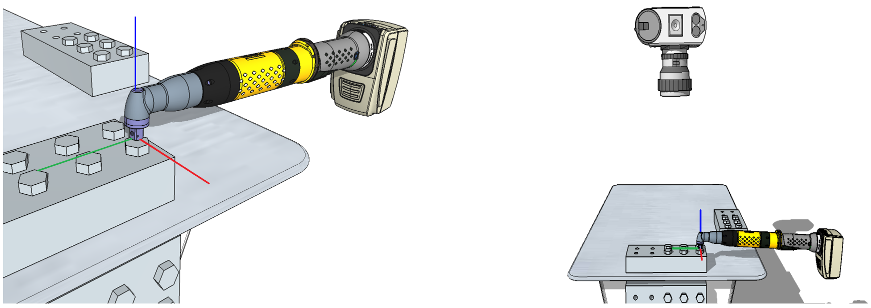
Figure 7: Concept 4

4.3 Testing

To be able to verify that the system meets the accuracy requirements, a two part process has been chosen. In order to facilitate rapid testing a simple test will first be used which will be further developed.

The first test concept is three or more blocks with threaded holes, see Figure 8a which have fixed geometry and will be machined with sub millimetre accuracy. Using a table and machined fixtures these boxes and a origin point are mounted in a special pattern to test a few common mounting scenarios taken from the industry. The origin point is a stand for the tool to be calibrated and reset. The gathered data will then be compared with the known geometry.

The second test concept is supported with a high-speed camera, see Figure 8b, for the ability to verify the tracking in 2D. These two data sets are synced with a led which is turned on from the switch and its state is also logged to the system collecting the test data. In this way testing of the human reaction time can also be verified.



(a) First test concept **(b)** 2D tracking

Figure 8: How to verify the position

4.4 Stakeholders

These are the identified stakeholders.

4.4.1 Stakeholder identification

- Atlas Copco
- The customers (Eg. Scania, Volvo)
- The operators
- The service technician

References

[1] D. G. Ullman, *The Mechanical Design Process.* McGraw-Hill, 2003.

[2] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *Robotics & Automation Magazine, IEEE*, vol. 18, no. 4, pp. 80–92, 2011.

[3] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1. IEEE, 2004, pp. I–652.

[4] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the mars exploration rovers," *Journal of Field Robotics*, vol. 24, no. 3, pp. 169–186, 2007.

[5] M. A. Mahammed, A. I. Melhum, and F. A. Kochery, "Object distance measurement by stereo vision," *International Journal of Science and Applied Information Technology (IJSAIT) Vol*, vol. 2, pp. 05–08, 2013.

[6] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision.* Cambridge university press, 2003.

[7] R. Mautz, "Indoor positioning technologies," Ph.D. dissertation, Habilitationsschrift ETH Zürich, 2012, 2012.

[8] L. Li, "Time-of-flight camera–an introduction," *Texas Instruments-Technical White Paper*, 2014.

[9] Mesa sr4500. [Online]. Available: http://www.mesa-imaging.ch/products/sr4500/ [Accessed 2015-05-14].

[10] Structured light. [Online]. Available: http://en.wikipedia.org/wiki/Structured_light [Accessed 2015-05-14].

[11] G. Csaba, L. Somlyai, and Z. Vámossy, "Mobile robot navigation in unknown environment using structured light," in *Logistics and Industrial Informatics (LINDI), 2011 3rd IEEE International Symposium on.* IEEE, 2011, pp. 249–254.

[12] Structured light accuracy. [Online]. Available: http://www.sci.utah.edu/~gerig/CS6320-S2012/Materials/CS6320-CV-S2012-StructuredLight.pdf [Accessed 2015-05-14].

[13] Laser measurement unit. [Online]. Available: http://www.faro.com/products/metrology/faro-laser-tracker/overview [Accessed 2015-05-15].

[14] Laser measurement unit. [Online]. Available: http://www.boschtools.com/Products/Tools/Pages/BoschProductDetail.aspx?pid=GLM%2015 [Accessed 2015-05-15].

[15] Laser triangulation. [Online]. Available: http://www.mtiinstruments.com/pdf/products/microtrak3.pdf [Accessed 2015-05-15].

[16] W. Zhang, M. I. S. Chowdhury, and M. Kavehrad, "Asynchronous indoor positioning system based on visible light communications," *Optical Engineering*, vol. 53, no. 4, p. 045105, 2014. [Online]. Available: http://dx.doi.org/10.1117/1.OE.53.4.045105

[17] S. Hann, J.-H. Kim, S.-Y. Jung, and C.-S. Park, "White led ceiling lights positioning systems for optical wireless indoor applications," in *Optical Communication (ECOC), 2010 36th European Conference and Exhibition on*, Sept 2010, pp. 1–3.

[18] Z. Zhou, M. Kavehrad, and P. Deng, "Indoor positioning algorithm using light-emitting diode visible light communications," *Optical Engineering*, vol. 51, no. 8, pp. 085 009–1–085 009–6, 2012. [Online]. Available: http://dx.doi.org/10.1117/1.OE.51.8.085009

[19] Y.-Y. Won, S.-H. Yang, D.-H. Kim, and S.-K. Han, "Three-dimensional optical wireless indoor positioning system using location code map based on power distribution of visible light emitting diode," *Optoelectronics, IET*, vol. 7, no. 3, pp. 77–83, June 2013.

[20] VLC positioning system. [Online]. Available: http://www.gelighting.com/LightingWeb/na/solutions/control-systems/indoor-positioning-system.jsp [Accessed 2015-05-15].

[21] S. J. P. M.M.G. Mazumder, S. Kim, "Precision and repeatablility analysis of optotrak certus as a tool for gait analysis utilizing a 3d robot," *Journal of Engineering and Technology Research*, vol. 3(2), 02 2011.

[22] Optotrak certus. [Online]. Available: http://www.ndigital.com/msci/products/optotrak-certus/ [Accessed 2015-05-12].

[23] H. A. A. J. L. Todor Stoyanov, Athanasia Louloudi, "Comparative evaluation of range sensor accuracy in indoor environments," *Robotics and Autonomous Systems*, vol. 61, 10 2013.

[24] S. O. E. K. Khoshelham, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, 12 2012.

[25] K. Audenaert, H. Peremans, Y. Kawahara, and J. van Campenhout, "Accurate ranging of mulitple objects using ultrasonic sensors," in *International Conference on Robotics and Automation*, vol. May. IEEE, 1992, pp. 1733–1738.

[26] H. E. Bass and L. N. Bolen, "Ultrasonic background noise in industrial environments," *The Journal of the Acoustical Society of America*, vol. 78, pp. 2013–2016, December 1985.

[27] S. Holm, "Hybrid ultrasound–rfid indoor positioning: Combining the best of both worlds."

[28] AMT ultrasonic triangulation. [Online]. Available: http://amt.alfing.de/en/products/handheld-tools/handheld-nutrunner.html [Accessed 2015-05-15].

[29] J. F. H. Wendeberg, S. C, and L. Reindl., "Anchor-free tdoa selflocalization," in *In Proceedings of IEEE IPIN 2011*. IEEE, 2011, p. 1–10.

[30] N. bodhi Priyantha, "The cricket indoor location system," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[31] M. Addlesee, R. Curwen, S. Hodges, J. Newman, P. Steggles, A. Ward, and A. Hopper, "Implementing a sentient computing system," in *Reprint from IEEE Computer*. IEEE, 2001, pp. 1–9.

[32] A. Harter, A. Hopper, P. Steggles, AndyWard, and P. Webster, "The anatomy of a context-aware application," *Wireless Networks*, vol. 1, pp. 1–16, 2001.

[33] C. Medina, J. C. Segura, and Ángel De la Torre, "Ultrasound indoor positioning system based on a low-power wireless sensor network providing sub - centimeter accuracy," *Sensors*, pp. 3501–3526, March 2013.

[34] Hexamite ultrasound. [Online]. Available: http://www.hexamite.com/ [Accessed 2015-05-15].

[35] Hexamite ultrasound hx19v3 product page. [Online]. Available: http://hexamite.com/hx19x.htm [Accessed 2015-05-18].

[36] H. liu, H. Darabi, P. Banerjee, and J. liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and reviews*, vol. 37, pp. 1067–1080, November 2007.

[37] J. Hightower, R. Want, and G. Borriello, "Spoton: An indoor 3D location sensing technology based on rf signal strength," *UW CSE 00-02-02, University of Washington, Department of Computer Science and Engineering, Seattle, WA*, vol. 1, 2000.

[38] D. Jana.Rovnakova, Maria.Svecova, "Signal processing for through wall moving target tracking by m-sequence uwb radar," *IEEE*, 2008.

[39] J. B. H. Chung, L. Ojeda, "Sensor fusion for mobile robot dead-reckoning with a precision-calibrated fiber optic gyroscope," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, 2001, pp. 3588 – 3593.

[40] A. A. M. Grewal, "Ask the experts: How good is your gyro?" *IEEE CONTROL SYSTEMS MAGAZINE*, pp. 12–14, 2010.

[41] Gyroscope primer. [Online]. Available: http://www.phidgets.com/docs/Gyroscope_Primer

[42] D. Vincent, "Accurate position tracking using inertial measurement units, white paper," *PNI Written in collaboration with Miami University*, 2013.

[43] A. D. King, "Inertial navigation – forty years of evolution," *GEC Review*, vol. 13, no. 3, pp. 140–149.

[44] Fundamentals of attitude estimation. [Online]. Available: http://www.chrobotics.com/library/attitude-estimation [Accessed 2015-05-17].

[45] M. Kraft, "Micromachined inertial sensors. the state of the art and a look into the future," *IMC Measurement and Control*, vol. 33, no. 6, pp. 164–168, 2000.

[46] R. Mautz, "Indoor positioning technologies," Ph.D. dissertation, ETH Zurich.

[47] E. A. Prigge, "A positioning system with no line-of-sight restrictions for cluttered environments," Ph.D. dissertation, Stanford University, 2005.

[48] G4 motion tracking. [Online]. Available: http://www.mesa-imaging.ch/products/sr4500/ [Accessed 2015-05-12].

[49] J. B. Sillero, "Sensor fusion methods for indoor navigation using uwb radio aided ins/dr," 07 2012.

[50] "Performance evaluation of multi-sensor fusion models in indoor navigation."

[51] T. Systems, "Neon 3D location - mission essential developers kit."

[52] S. Tilch and R. Mautz, "Development of a new laser-based, optical indoor positioning system," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences Commission*, vol. 1501, pp. 575–580, 2010.

A Appendix

Technique		Accuracy	Difficulty of manipulation of system	Update freq >10Hz	Pos lag of sys <200ms	6 degrees of freedom positioning	Pos tool <500 mm accuracy
Vision	Camera on Tool	2 cm	Motion blur will occur with extensive movement.	Can probably be met.	Can probably be met.	Yes	Yes
	Sterio Vision	Less than 5cm, can be very good	Targets may be blocked by operators	Yes	Can probably be met.	Yes	Yes
	Time of Flight Camera	2 cm	Motion blur, blocked by some objects	Can probably be met.	Can probably be met.	Yes	Yes
	Structured Light	Less than 0.01mm	Inaccuracy wit a lot of movement	Can probably be met.	Can probably be met.	Yes	Yes
Light	Laser - Time of Flight	~0.5 cm, can be 0.009 mm	Can cover the sensor.	Yes	Yes	No, 1 dimension measurement	Not alone
	Laser - Triangulation	0.04 µm	Can cover the sensor	Yes	Yes	No, 1 dimension measurement	Not alone
	Projected IR light	0.1 mm	Can cover the sensor	Yes	Yes	Yes with 3 markers	Yes
	Visable Light Communication	~0.6 cm	Can cover the sensor	Yes	Yes	Hard with sensor placement	Yes
Sound	Time Difference of Arrival	2.5 cm	Maybe by covering the transmitter	Yes	Yes	No, only position	Yes
	Time of Flight	0.2 - 2 cm	Hard/Medium	Yes	Yes	No, only position	Yes
	Angle of Arrival - AMT Ultrasonic	"Very exact"	Hard	Yes	Yes	No, only position	Yes
Radio	Ultrawideband	Best 2cm (line of sight), avg 15cm	Hard	Yes	Yes	Possble	Yes
Magnetic	Artifical AC magnetic field	Few milimeter	Medium	Yes	Yes	Yes	Yes
Hybrid Systems	UWB, IMU & dead reckoning	4-5 cm (simulation) / ~40 cm (experiment)	Hard	This can probably be met.	This can probably be met.	2dof, can probably be extended	Yes
	CLIPS	Deviation is about 2 mm/m	Medium	This can probably be met.	This can probably be met.	Yes, with line of sight	Yes
	FARO	0.009 mm	The reflector may be covered.	Yes	Yes	Yes with IMU, requires LoS	Yes
IMU	Inertial Measurement Unit	0.3 deg/root-hour drift for gyroscopes. 0.1-1 Accelerometer bias. Strategic grade IMU ~30 m/h drift. Navigation graded gyroscopes 0.005 deg/root-hour drift.	If the unit is shaked heavily, the accelerometer will get very incorrect readings.	This can probably be met.	This can probably be met	Yes	Timedependent, see accuracy

Price	Robustness in industrial environment	Size of system	Weight of system	Intrusiveness		Technique
50 $	Low			Intrusive	Camera on Tool	Vision
Depends on cameras	Good	Can be big	Can be heavy	Intrusive	Sterio Vision	
4500 $	Low			Intrusive	Time of Flight Camera	
500 SEK	Medium	Can be large or can be small on the tool		Not intrusive	Structured Light	
500 SEK, can be more	Good	32x27x15 mm	<20g	Not intrusive	Laser - Time of Flight	Light
Expensive	Good	76x74x26 mm		Not intrusive	Laser - Triangulation	
Expensive	Decent	Markers is only led		Not intrusive	Projected IR light	
~1000 SEK	Decent	Size of photo sensor		Not intrusive	Visable Light Communication	
Unknown, not very expensive	Medium/Good	Small (size of speaker)	Lightweight ~200 g	Not intrusive	Time Difference of Arrival	Sound
1500-2000 EUR for Hexamite system	Medium/Good	Roughly the size of matchbox	~200g	Not intrusive	Time of Flight	
Unknown	Medium/Good	Small (size of a speaker)	~50g	Not intrusive	Angle of Arrival - AMT Ultrasonic	
Unknown	Medium/Good	Ubisense - 83 x 42 mm, PulsON - 76 x 80mm	PulsOn - 58 g	Not intrusive	Ultrawideband	Radio
Unknown	Weak/Medium	~ 15x15x15 mm	Assume ~100g	Not intrusive	Artifical AC magnetic field	Magnetic
Ubisense + IMU	Low	Ubisense + IMU	Low	Not intrusive	UWB, IMU & dead reckoning	Hybrid Systems
Unknown, not very expensive	Medium	Camera + Lasers	Low on the tool	Might be intrusive	CLIPS	
Very expensive	Very good	Approximately a sphere with a 1 cm radius	<50 g	Might be intrusive	FARO	
100 – 5000 SEK Around 100 000 SEK for military grade gyroscopes (very large).	The unit is only affected by shaking, no ither element will affect it	They can be very large, however, we can easily find IMUs in the size-range we would need	They can be heavy, but we can easily fing lighter ones	None	Inertial Measurement Unit	IMU

B Appendix

	Elin			Satyajit			Linxuan			Stefan			Mathias		
	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief
Camera on Tool	48,7	45,0	25,2	56,2	43,3	24,0	52,8	34,8	16,8	62,3	56,0	35,1	32,8	34,5	12,1
Sterio vision	69,2	51,0	36,0	66,8	58,2	38,2	62,7	48,5	29,8	56,5	50,5	29,7	82,0	58,5	51,8
TOF Camera	50,3	47,3	22,2	66,3	56,5	39,6	56,9	48,1	25,7	55,0	57,6	33,4	42,6	29,6	18,7
Structured light	51,2	35,7	18,1	61,7	55,9	36,4	56,0	42,8	23,1	51,9	47,7	26,7	30,9	27,7	11,8
Time of flight laser	62,8	42,2	27,7	65,6	41,9	28,7	65,6	47,0	32,4	62,7	64,5	44,3	44,0	32,6	18,0
Triangulation Laser	60,5	51,8	29,8	58,2	31,4	20,1	70,1	47,2	35,1	62,4	56,8	39,6	37,4	30,2	13,2
Projected IR light	67,4	32,8	25,1	67,4	40,2	27,9	73,4	40,7	32,2	76,4	67,0	54,4	23,2	15,5	4,1
Kinect	57,4	37,5	21,4	51,3	62,9	32,7	60,4	41,5	24,6	60,2	60,6	38,0	45,9	41,0	23,6
Visable Light com	65,3	39,0	27,9	66,5	35,7	25,2	68,1	40,4	28,1	46,7	54,0	28,8	32,1	22,0	9,8
ToF	64,2	43,7	27,9	69,2	63,4	44,1	52,2	47,8	25,4	68,6	63,6	45,7	57,7	54,4	33,3
Triangulation and AoA	50,4	33,8	20,2	73,2	66,0	48,5	53,5	43,3	23,9	68,2	60,0	42,6	53,2	40,6	23,6
TDoA	43,1	30,7	13,8	69,5	67,9	47,4	52,6	49,4	26,2	60,1	59,1	38,2	45,7	40,3	21,4
Ultrawideband	58,5	29,6	18,8	79,0	75,2	59,9	53,4	43,5	22,6	57,2	61,7	37,9	81,3	56,9	46,6
Artifical AC magnetic field	68,2	36,5	25,1	60,3	22,4	14,0	68,3	36,1	25,2	75,4	70,4	56,8	33,5	17,0	7,0
UWB, IMU & dead reckoning	49,0	30,0	14,1	72,1	42,7	30,9	56,5	37,8	20,9	65,3	58,4	41,5	79,3	50,2	39,9
Camera and laser	63,7	23,1	17,2	66,5	41,6	27,5	56,0	41,2	23,6	65,6	55,4	38,5	46,4	43,5	20,5
FARO	71,4	40,0	29,0	71,2	37,2	27,2	67,1	41,5	29,2	76,1	72,3	59,3	24,0	16,4	4,7
		Avg Belief	23,5		Avg Belief	33,7		Avg Belief	26,2		Avg Belief	40,6		Avg Belief	21,2

	Michael			Björn			Tobias			Adam			Christoffer		
	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief	Confidence	Knowledge	Belief
Camera on Tool	44,3	60,3	24,4	49,7	50,4	25,6	42,3	42,9	19,0	41,6	39,0	17,0	43,1	46,2	21,7
Sterio vision	80,6	68,2	55,8	64,4	57,1	37,3	34,7	42,9	16,3	60,7	52,9	32,4	61,4	52,5	33,6
TOF Camera	49,2	52,8	27,7	56,9	49,4	27,6	43,0	42,9	19,1	52,6	35,1	19,3	63,9	50,5	32,9
Structured light	45,8	50,6	23,9	51,7	41,7	22,2	43,6	42,9	19,4	56,3	44,8	26,9	44,4	31,6	14,1
Time of flight laser	56,0	55,7	28,9	58,5	49,5	30,0	62,3	63,4	40,7	68,1	55,0	38,4	62,7	52,1	34,3
Triangulation Laser	56,2	56,6	29,4	56,8	54,6	30,0	59,8	60,8	37,7	65,9	55,2	36,7	64,1	50,7	34,1
Projected IR light	74,9	39,5	29,3	72,6	44,5	33,9	80,0	48,9	39,1	73,2	55,3	40,5	46,5	26,6	12,5
Kinect	74,3	58,6	45,2	56,2	45,5	26,6	66,7	52,2	35,3	58,7	53,7	31,8	55,0	55,3	30,6
Visable Light com	75,8	59,0	46,3	70,2	45,8	32,9	63,8	59,9	38,2	71,5	60,2	43,6	45,9	23,1	10,8
ToF	80,4	60,4	49,4	72,4	46,6	34,1	71,3	45,9	32,6	79,8	80,4	64,8	59,6	50,9	30,4
Triangulation and AoA	79,1	60,4	48,7	62,9	38,7	26,3	72,5	43,5	31,3	75,0	76,4	57,8	61,1	46,1	28,3
TDoA	78,7	60,8	48,6	52,4	39,4	24,1	65,4	45,9	29,6	62,7	68,8	46,1	64,0	42,6	28,0
Ultrawideband	72,0	91,5	66,4	68,0	57,6	41,6	65,8	53,2	35,4	67,5	52,2	40,9	61,1	50,0	31,1
Artifical AC magnetic field	78,6	61,8	50,1	71,4	44,8	33,4	70,5	42,2	30,0	67,5	49,4	34,2	58,5	27,7	16,7
UWB, IMU & dead reckoning	56,7	68,1	33,4	50,1	50,4	25,8	68,2	47,5	33,3	57,6	48,5	27,9	63,8	40,9	26,5
Camera and laser	54,2	28,3	14,7	59,4	27,6	17,3	65,6	31,6	20,9	65,9	40,6	28,5	73,8	67,6	50,9
FARO	70,6	46,4	36,8	66,4	38,7	26,8	77,4	59,2	46,1	71,8	45,9	35,7	64,5	34,5	22,6
		Avg Belief	38,8		Avg Belief	29,1		Avg Belief	30,8		Avg Belief	36,6		Avg Belief	27,0

C Appendix

	Accuracy	Difficulty of manipulation of system	Update freq >10Hz	Pos lag of sys <200ms	Pos tool <500 mm accuracy in workspace	Cost	Robustness in industrial environment	6 dof positioning	size of system	weight	Intrusiveness
Sum	**23,75**	**7,75**	**8,625**	**7,625**	**9**	**1,375**	**15**	**13,125**	**4,875**	**3,625**	**5,25**
Stefan	30	5	10	9	5	1	15	20	2	2	1
Elin	25	8	10	10	20	1	10	10	2	2	2
Satyajit	20	8	2	4	1	1	20	20	8	8	8
Linxuan	20	10	12,5	7,5	10	2,5	15	2,5	7,5	2,5	10
Mathias	30	8	15	14	5	2	11	7	1	2	5
Michael	25	5	3	5	5	7	21	22	3	3	1
Björn	20	3	10	15	10	5	15	15	3	3	1
Tobias	15	15	1	15	1	12	20	10	1	5	5
Adam	25	6	14	7	10	8	4	20	3	2	1
Christoffer	30	5	8	4	4	5	8	20	7	5	4

B Requirements

1 Requirements

1.1 Functional Requirements

Accuracy:

- The system shall be able to measure the central position of the outgoing tap of the tool with accuracy of 50 mm when at a distance of 600 mm or closer to the target.
- The system shall be able to measure the central position of the outgoing tap of the tool with an accuracy of 500 mm inside a work space.

Output:

- The system shall update the position of the tool with a frequency of at least 10 Hz.
- The system shall output the position of the tool in the form of a Cartesian coordinate system in three-dimensional space (X, Y, Z) and the normal vector of the outgoing tap.
- The origin of the Cartesian coordinate system shall be defined as a fixed point in the work space.
- The systems position lag shall not be greater average human reaction time, estimated to 200 milliseconds.

Robustness:

- The system shall be robust enough to operate in an simulated[1] industrial environment with a limited line of sight[2] without failing the accuracy requirements.

1.2 Soft requirements

Accuracy: The system shall be able to measure the central position of the outgoing tap of the tool with an accuracy of 2000 mm when located in a global indoor positioning system.

Intrusiveness: The worker should not feel supervised and/or tracked.

[1] A lab created approximation of an industrial environment containing metal objects and including disturbances from the tool itself.

[2] Objects that partly blocks the line-of-sight between tool and receiver.

Robustness:

- The system should be robust enough to not create a stop in the production.
- The system should not be manipulable by the operator.

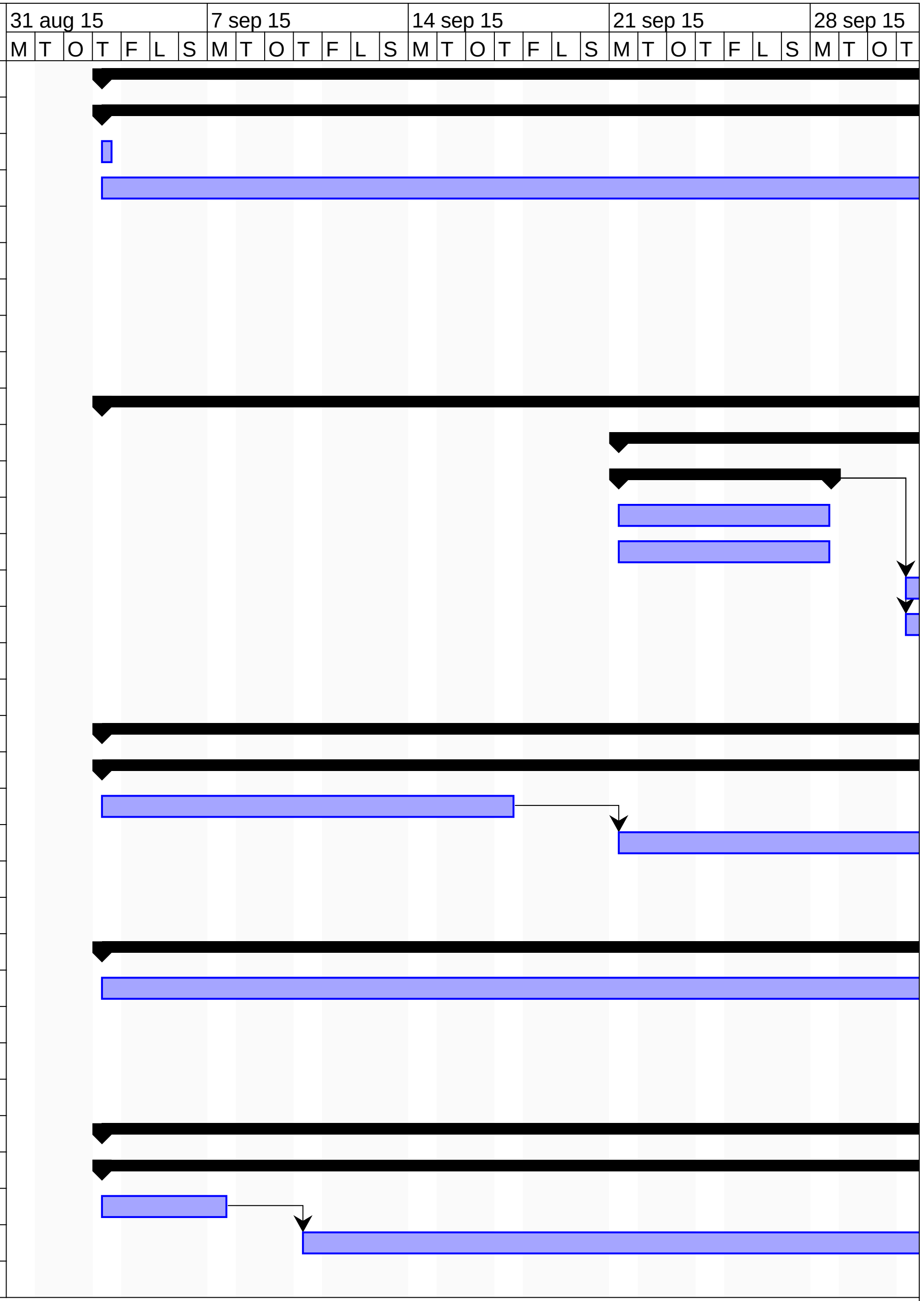
Cost: The cost of the tracking system is important however it is not a constraint for the state of the art.

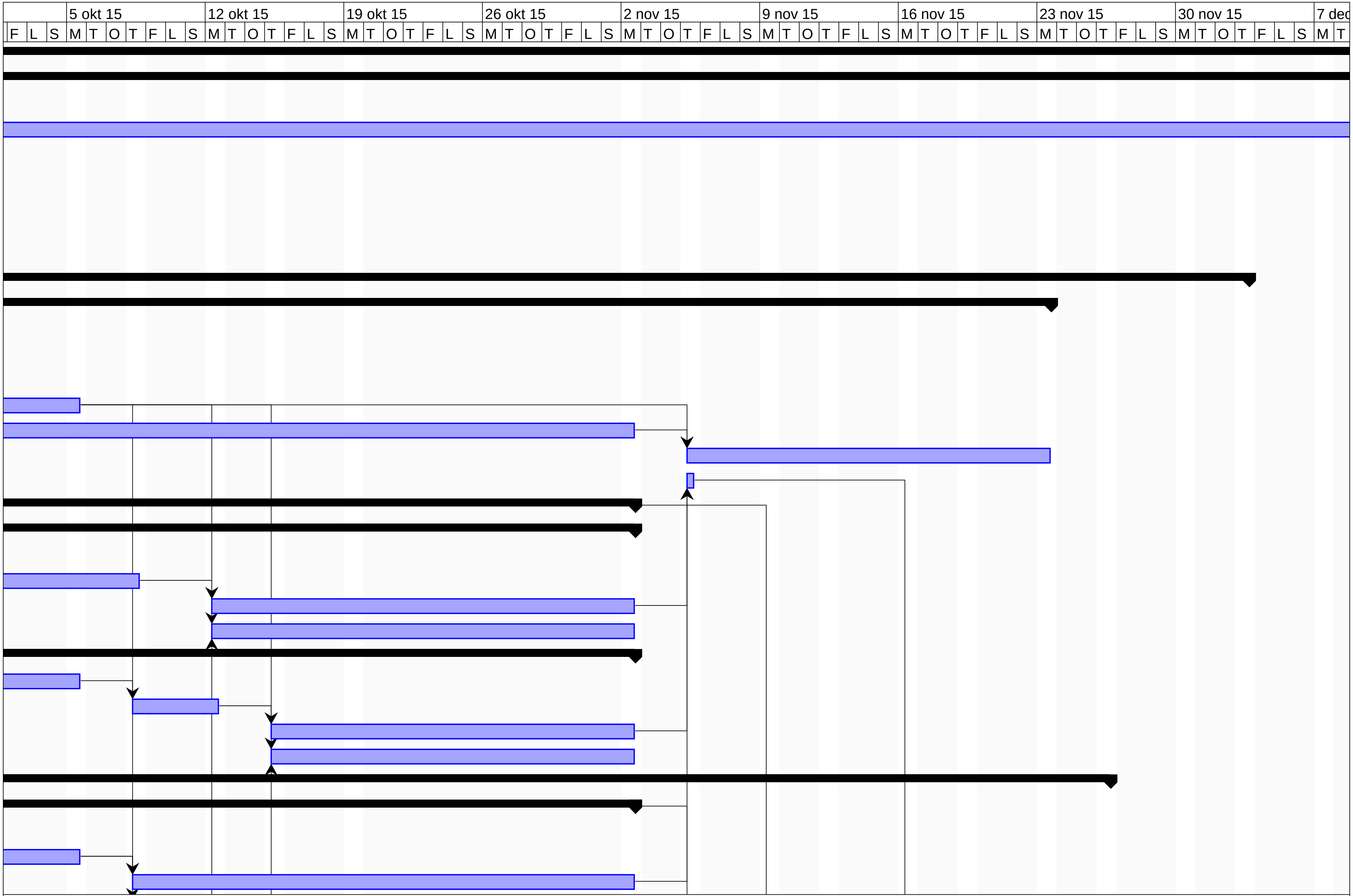
Scalability: The system should be easy to install/remove/change, however it is not a constraint for the state of the art.

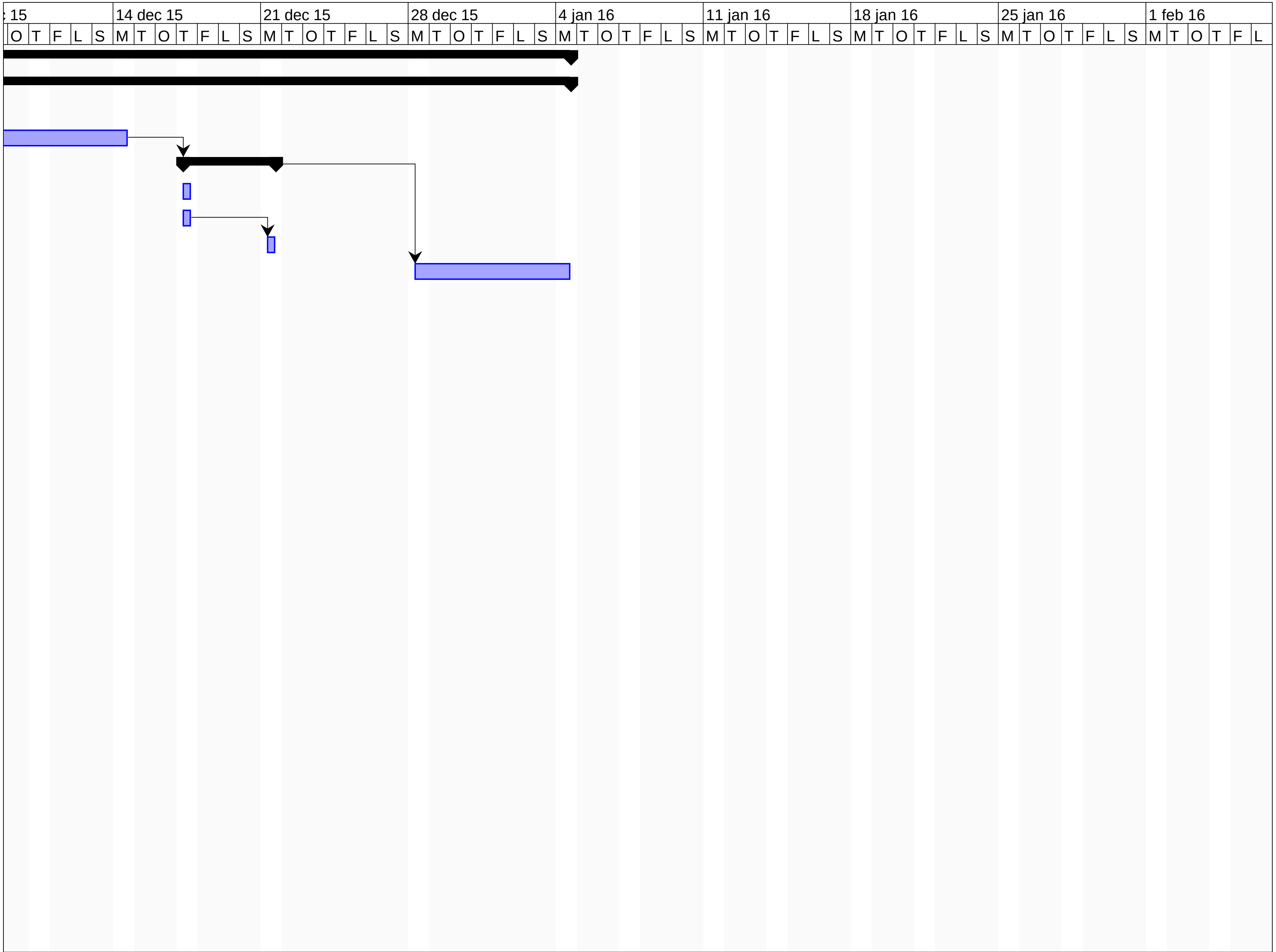
Dimension: A possible add-on to the tool should not affect the usage significantly. The size and weight of the add-on should preferably be of similar dimensions as the current add-ons.

C Planning

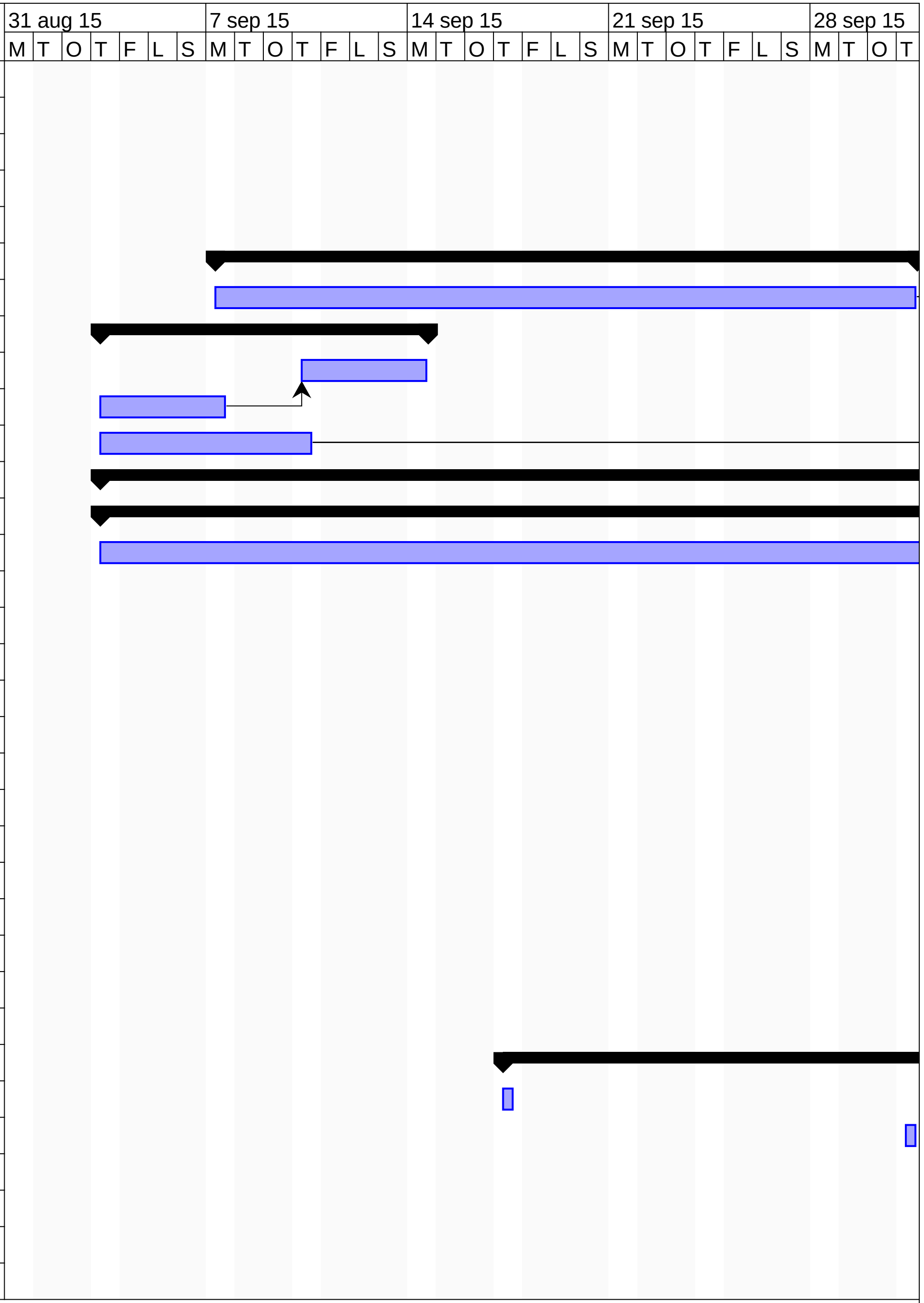
		Namn	Varakti...	Föreg...	Start	Slut
1		**Project**	**31 da...**		**2015...**	**2016-01-0...**
2		**Report**	**31 da...**		**2015...**	**2016-01-0...**
3		Structure	1 dag		2015-...	2015-09-03 ...
4		Content	27 da...		2015-...	2015-12-14 ...
5		**Format**	**2 dag...**	**4**	**2015...**	**2015-12-2...**
6		Pictures	1 dag		2015-...	2015-12-17 ...
7		Language	1 dag		2015-...	2015-12-17 ...
8		Abbreviations	1 dag	7	2015-...	2015-12-21 ...
9		Proofreading	2 dagar	5	2015-...	2016-01-04 ...
10		**Development**	**24 da...**		**2015...**	**2015-12-0...**
11		**Platform**	**16 da...**		**2015...**	**2015-11-2...**
12		**Overall design**	**3 dag...**		**2015...**	**2015-09-2...**
13		Functionality desicion	3 dagar		2015-...	2015-09-28 ...
14		GUI desicion	3 dagar		2015-...	2015-09-28 ...
15		Function interface documentation	2 dagar	12	2015-...	2015-10-05 ...
16		Function internal platform documenta...	7 dagar	12	2015-...	2015-11-02 ...
17		GUI	6 dagar	15;16	2015-...	2015-11-23 ...
18		Testing data collection	1 dag	23;2...	2015-...	2015-11-05 ...
19		**Base**	**15 da...**		**2015...**	**2015-11-0...**
20		**UWB**	**15 da...**		**2015...**	**2015-11-0...**
21		Ordering/Aquire	5 dagar		2015-...	2015-09-17 ...
22		Get up and running	6 dagar	21	2015-...	2015-10-08 ...
23		Make platform function	4 dagar	15;22	2015-...	2015-11-02 ...
24		Testing	4 dagar	22;45	2015-...	2015-11-02 ...
25		**IMU**	**15 da...**		**2015...**	**2015-11-0...**
26		Get up and running	10 da...		2015-...	2015-10-05 ...
27		Implement filters	2 dagar	26	2015-...	2015-10-12 ...
28		Make platform function	3 dagar	15;27	2015-...	2015-11-02 ...
29		Testing	3 dagar	45;27	2015-...	2015-11-02 ...
30		**Sys 1**	**22 da...**		**2015...**	**2015-11-2...**
31		**Steriovision**	**15 da...**		**2015...**	**2015-11-0...**
32		Ordering/Aquire	2 dagar		2015-...	2015-09-07 ...
33		Get up and running	8 dagar	32	2015-...	2015-10-05 ...
34		Make platform function	5 dagar	15;33	2015-...	2015-11-02 ...

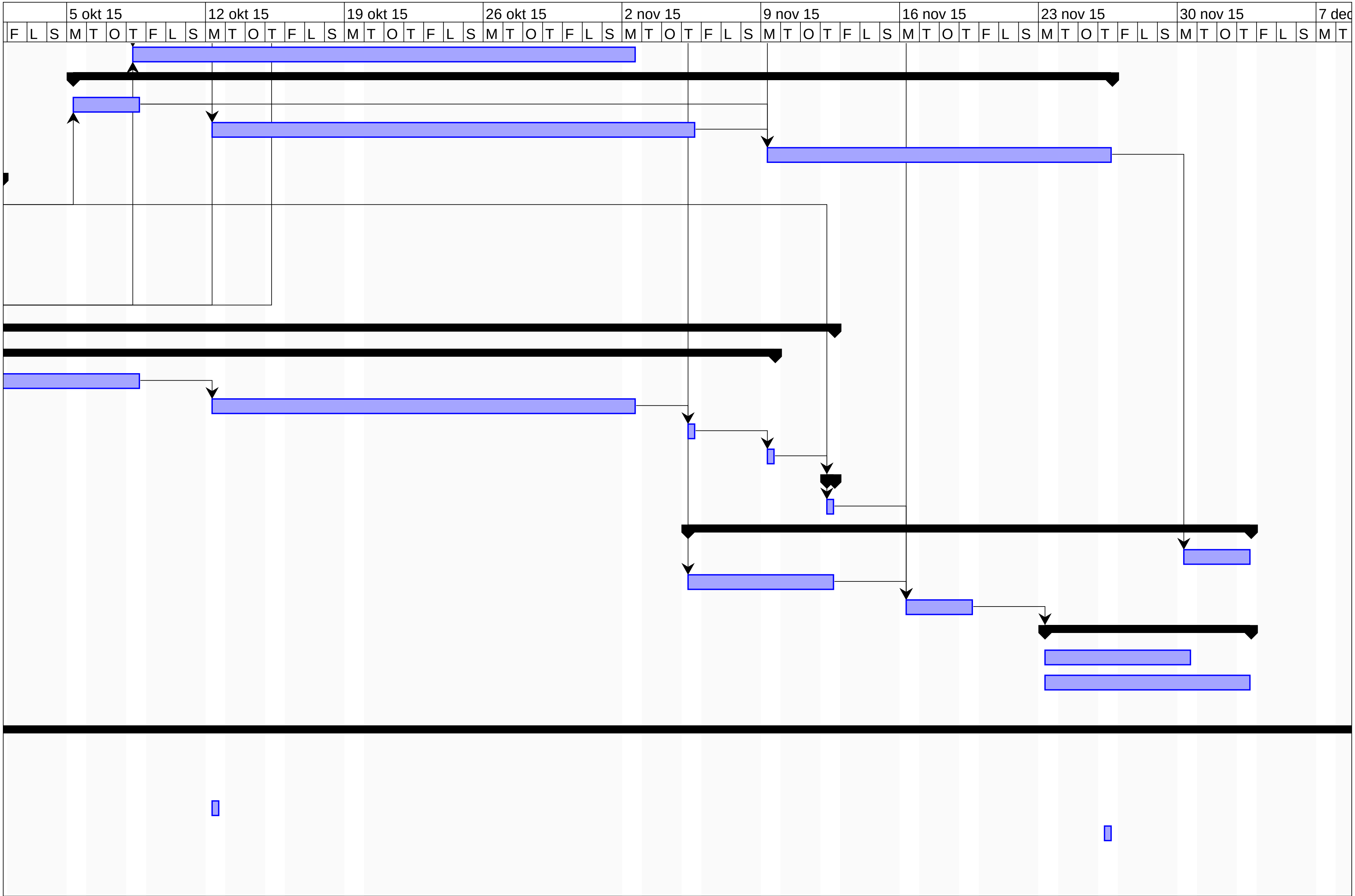


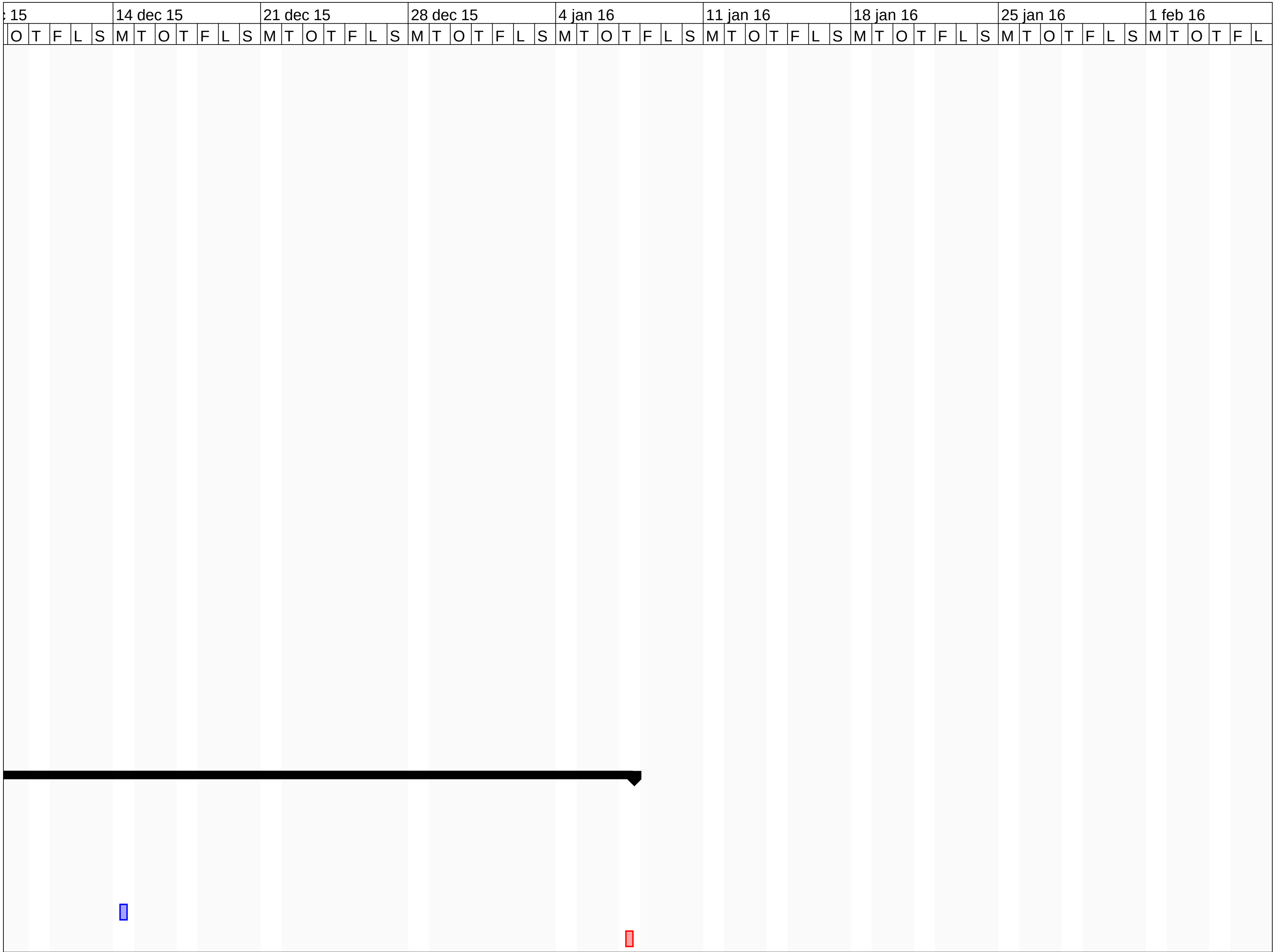

5 okt 15
12 okt 15
19 okt 15
26 okt 15
2 nov 15
9 nov 15
16 nov 15
23 nov 15
30 nov 15
7 dec
F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T


14 dec 15
21 dec 15
28 dec 15
4 jan 16
11 jan 16
18 jan 16
25 jan 16
1 feb 16

		Namn	Varakti...	Föreg...	Start	Slut
35		Testing	5 dagar	33;45	2015-...	2015-11-02 ...
36		**Sensor Fusion**	**13 da...**		**2015...**	**2015-11-2...**
37		Fusion Plan	2 dagar	41	2015-...	2015-10-08 ...
38		Sensor fusion shell functions	5 dagar	15;37	2015-...	2015-11-05 ...
39		Steriovision + UWB + IMU	6 dagar	19;3...	2015-...	2015-11-26 ...
40		**Sensor Fusion**	**8 dag...**		**2015...**	**2015-10-0...**
41		Theory	8 dagar		2015-...	2015-10-01 ...
42		**Test plan**	**4 dag...**		**2015...**	**2015-09-1...**
43		Testing-rigg build	2 dagar	44	2015-...	2015-09-14 ...
44		Testing design	2 dagar		2015-...	2015-09-07 ...
45		Testing schedule	3 dagar		2015-...	2015-09-10 ...
46		**Sys 2**	**18 da...**		**2015...**	**2015-11-1...**
47		**Ultrasound**	**17 da...**		**2015...**	**2015-11-0...**
48		Ordering/Aquire	11 da...		2015-...	2015-10-08 ...
49		Get up and running	4 dagar	48	2015-...	2015-11-02 ...
50		Implement filters	1 dag	49	2015-...	2015-11-05 ...
51		Testing	1 dag	50	2015-...	2015-11-09 ...
52		**Sensor Fusion**	**1 dag**	**41**	**2015...**	**2015-11-1...**
53		Ultrasound + UWB + IMU	1 dag	51	2015-...	2015-11-12 ...
54		**Final Concept**	**9 dag...**		**2015...**	**2015-12-0...**
55		Overall Design	2 dagar	39	2015-...	2015-12-03 ...
56		Prototype	3 dagar	31	2015-...	2015-11-12 ...
57		Testing	2 dagar	18;5...	2015-...	2015-11-19 ...
58		**Conclusion**	**4 dag...**	**57**	**2015...**	**2015-12-0...**
59		Future work/development future	3 dagar		2015-...	2015-11-30 ...
60		Discussion	4 dagar		2015-...	2015-12-03 ...
61		**Gates**	**28 da...**		**2015...**	**2016-01-0...**
62		DR 1	1 dag		2015-...	2015-09-17 ...
63		DR 2	1 dag		2015-...	2015-10-01 ...
64		MT	1 dag		2015-...	2015-10-12 ...
65		DR4	1 dag		2015-...	2015-11-26 ...
66		FP	1 dag		2015-...	2015-12-14 ...
67		Report hand in	1 dag		2016-...	2016-01-07 ...




5 okt 15
12 okt 15
19 okt 15
26 okt 15
2 nov 15
9 nov 15
16 nov 15
23 nov 15
30 nov 15
7 dec
F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T



15
14 dec 15
21 dec 15
28 dec 15
4 jan 16
11 jan 16
18 jan 16
25 jan 16
1 feb 16
O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L S M T O T F L

D SVN

Together with the report, the file-folder used for this project is included. The folder is including all project files such as meeting protocols, code, Matlab-files, research information and part-time summaries of the work together with the pre-study and the final report.

E Meetings

All the meeting protocols are stored in the folder under SVN\meetings, in this appendix the referenced meetings are listed and a short summary of them are included. The meetings are named the same as the file which are named according to the template "date, meeting number and name - author".

E.1 20150901 Design Decision Meeting - BM

During this meting the group went through the pre-study report one individual technique by one. Combinations of systems was considered and evaluated against each other.

E.2 20150902 Meeting 5 with Atlas - CL

During this meting with Atlas Copco, the decision was made to use existing positioning systems instead of building one from scratch. This decision was made because the accuracy is an important aspect of the project.

E.3 20150917 Meeting 7 with Atlas - AE

Atlas Copco put value on modularity and think it is good to focus on in the project, and likes the idea of the software platform. Atlas also looked over the financial status of the project.

E.4 20150924 Meeting with Sagar Behere

Talked about how to design the architecture for the platform. It is a trade-off between computation power and how close to Real time we will be able to get. Using a regular computer with Linux installed could deliver close to real time. But not hard real time, makes the kernel preemptive with some settings. Decision should consider: Computational power needed, how much real time needed and the group members knowledge. Decision ladder for choosing CPU:

- 1. Regular computer with Linux (core i7 or similar)
- 2. Beagle Bone Black running Linux
- 3. Micro controller (e.g. STM32F4, should have sufficient computational power for our use. Sagar has used this for a balancing of a Segway-style robot with 5 9DOF IMUs.)

- 4. Arduino

E.5 20151104 Meeting 9 with Atlas – AE

Presented and decided together with Atlas on the new approach of the system that focus on sensor fusion testing with three of the systems. Stereo vision, UWB and IMU. Ultrasound is set on pause, due to lack of time.

F Financials

Tool Tracker orders

Ultrasound system

Item	Orderowner	Date	Atlas approver	Date	Ordered	Delivered	SEK	Note	OK from Atlas
Ultrasound system	Mattias Nilsson	20150909	Stefan Olofsson	20150910	20151002	20151002	15 495,83 kr ?	20151002 -E-mail - Payment to Hexamite Ltd - MN	
						Sum	15 495,83 kr		
						Budget	15 000,00 kr		20150910 - EMAIL ATLAS - Purchase of system and material for test rig – AE
						Balance	-495,83 kr		

UWB system

Item	Orderowner	Date	Atlas approver	Date	Ordered	Delivered	SEK	Note	OK from Atlas
UWB system	Tobias Lundin	20150909	Stefan Olofsson	20150910		20150917	13 635,99 kr		
Powersupply for UWB system	Björn Magnusson	20150917					480,80 kr		
J-tag	Björn Magnusson						261,60 kr		
Extension power cable	Tobias Lundin	20151101				20151101	199,80 kr		
						Sum	14 578,19 kr		
						Budget	15 000,00 kr		20150910 - EMAIL ATLAS - Purchase of system and material for test rig – AE
						Balance	421,81 kr		

Other

Item	Orderowner	Date	Atlas approver	Date	Ordered	Delivered	SEK	Note	OK from Atlas
IMUs	Tobias Lundin	20150504	Stefan Olofsson	20150428	20150526	20150907	414,00 kr		2015-04-28-Conversation-E-mail-with-Atlas-AE
Test Rig materials	Björn Magnusson	20150907	Stefan Olofsson	20150910	NaN	20150907	347,16 kr	Kundorderbekräftese (might add paint cost)	20150910 - EMAIL ATLAS - Purchase of system and material for test rig – AE
Share Latex	Björn Magnusson	20150903	Stefan Olofsson	20150904	20150907	NaN	550,00 kr ?		20150925 0909PM - EMAIL ATLAS - Accuracy requirement and ok for Latex costs – AE
2 Arduinos + 2 usb-cables	Adam Ekström	20151029	Stefan Olofsson	20151022	20151027	20151030	831,49 kr		20151022 1659PM - EMAIL ATLAS - Non disclosure and ok for arduinos - AE
Markers for Stereo Vision	Adam Ekström	20151119	Stefan Olofsson	20151119	20151119	20151123	280,00 kr		20151119 1252 - EMAIL ATLAS - OK markers - AE
Material 3Dprinter OnTool Add-	Mattias Nilsson	20151126			20151126		150,00 kr		20151119 1348 - EMAIL ATLAS - Weekly financial status OK - AE
Clear Varnish Spray (Klarlack)	Elin Nordmark	20151201			20151201	NaN	99,00 kr		20151119 1348 - EMAIL ATLAS - Weekly financial status OK - AE
						Sum	2 671,65 kr		
						Budget	20 000,00 kr		
						Balance	17 328,35 kr		

Donations

Item	Orderowner	Date	Atlas approver	Date	Ordered	Delivered	SEK	Note	OK from Atlas
Stereovision system							100 000,00 kr	Lent by the department	
IMUs							5 900,00 kr	Given by Atlas Copco.	
						Sum	105 900,00 kr		

Total sum	32 745,67 kr
Total budget	50 000,00 kr
Total balance	17 254,33 kr

G Schematics

Schematics Test Rig

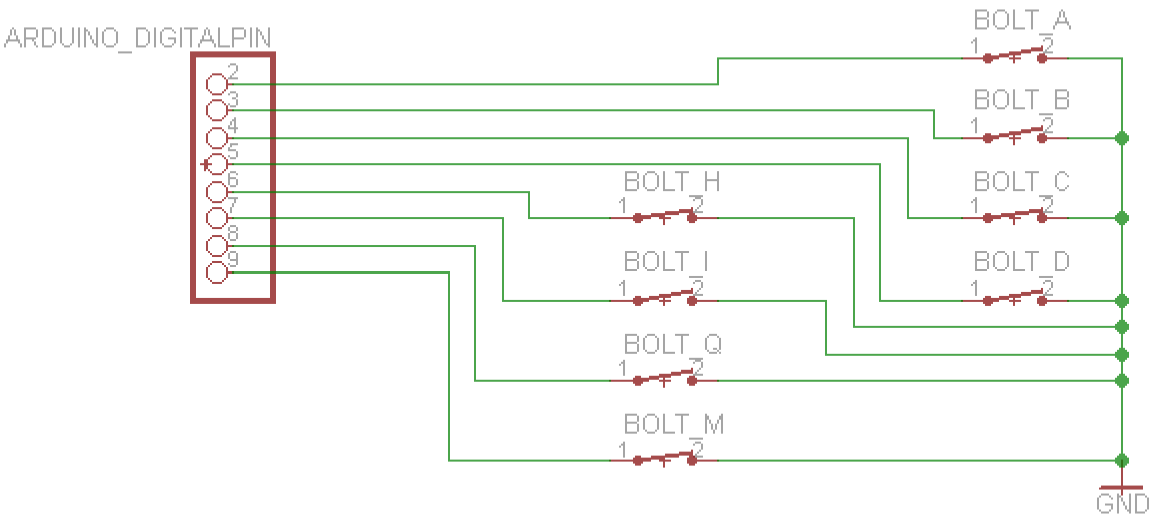


Figure 18: Bolt sensors circuit diagram.

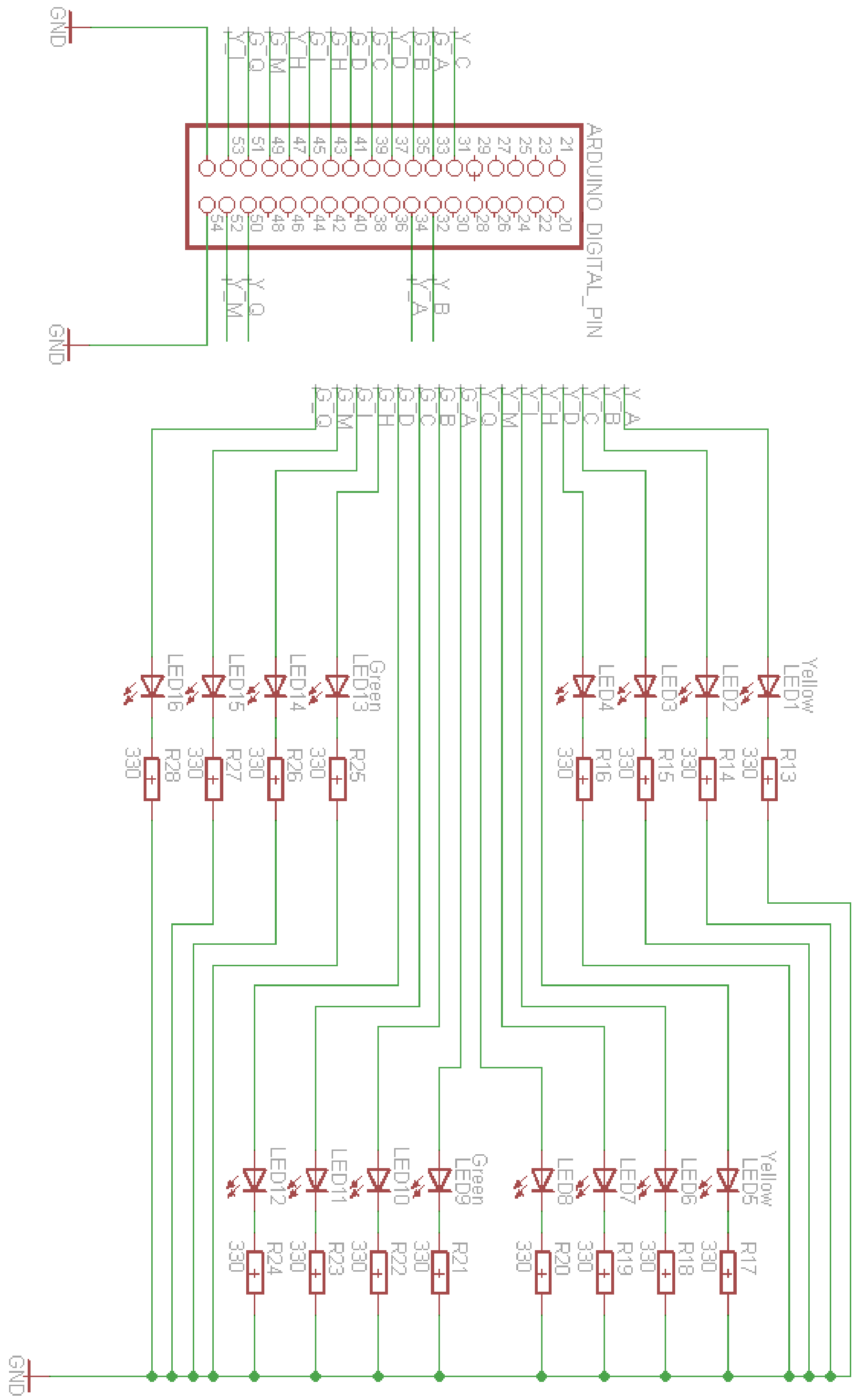


Figure 19: LED board circuit diagram.

Schematics On Tool

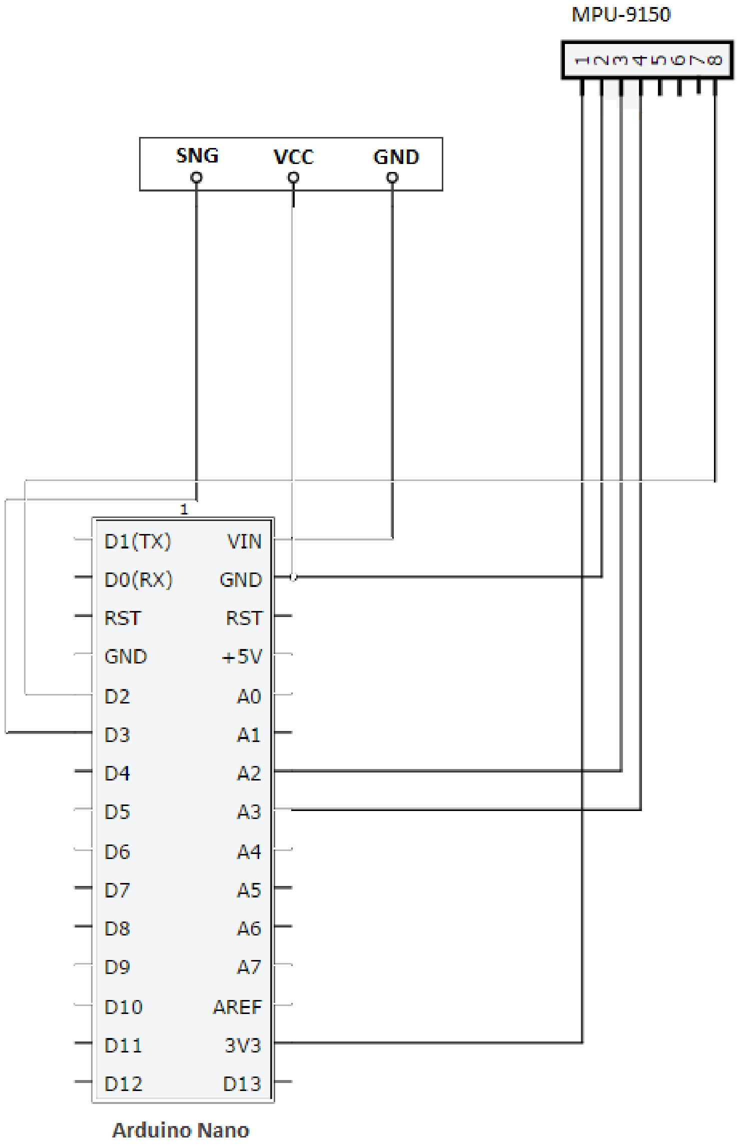
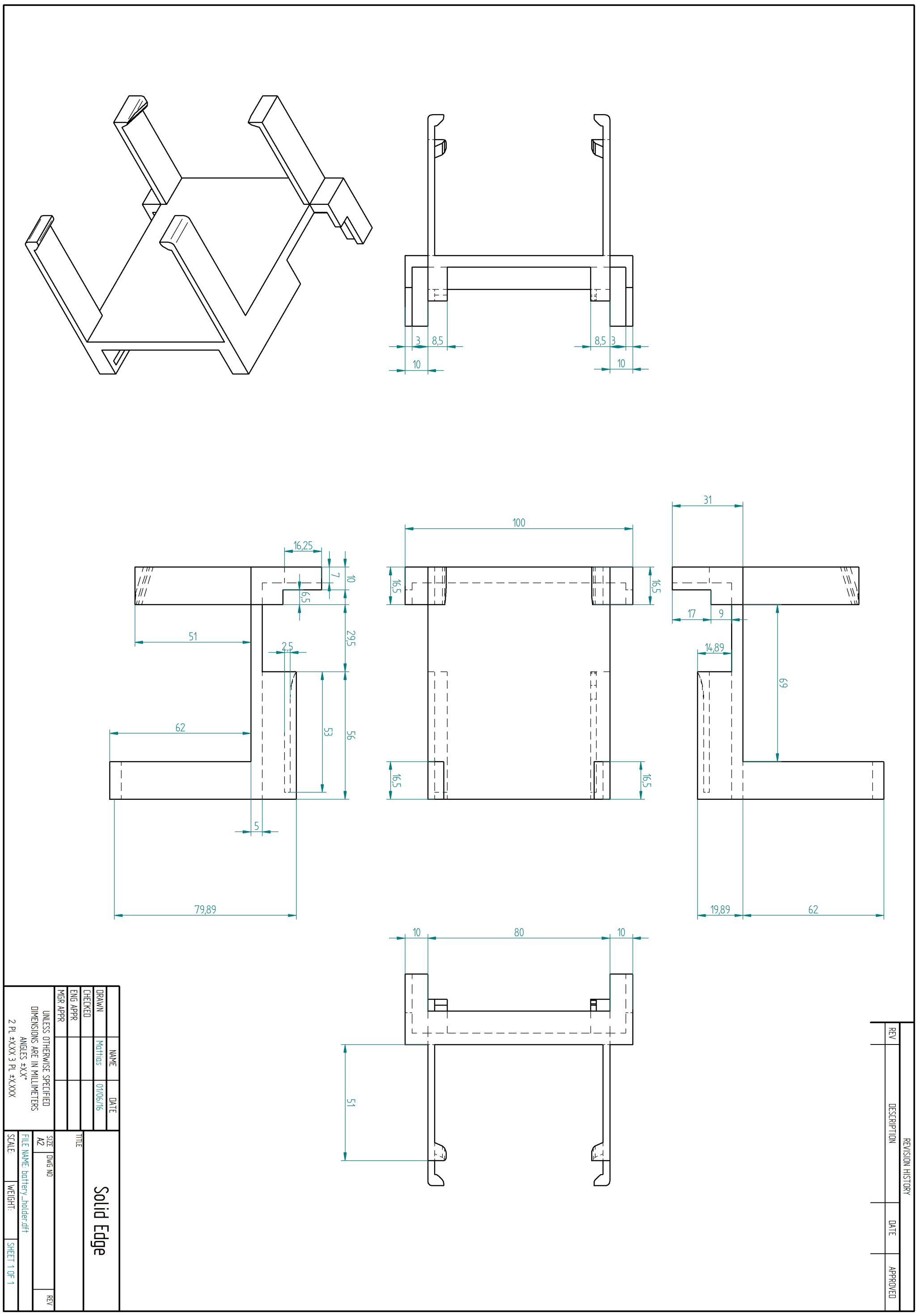
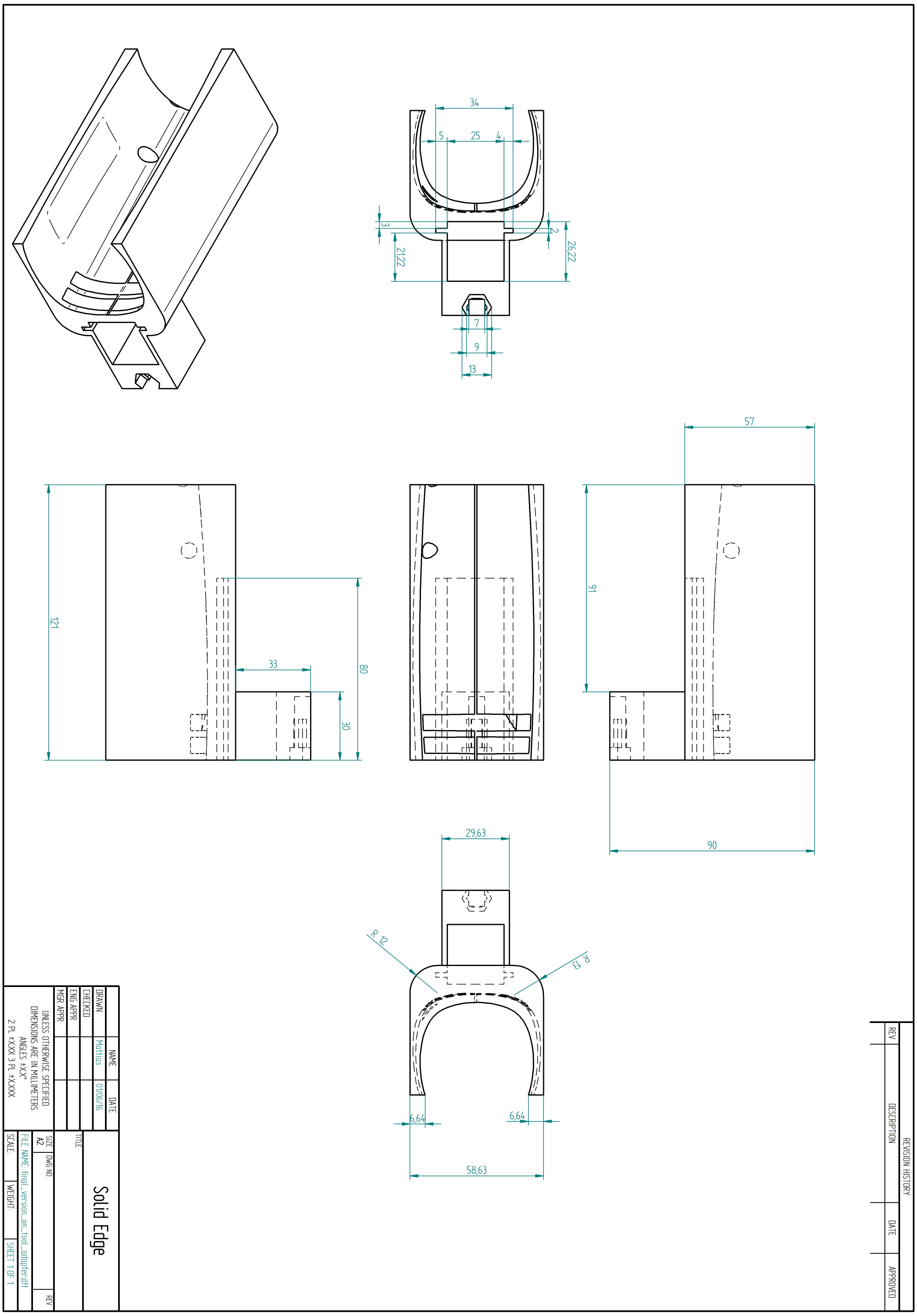


Figure 20: On tool circuit diagram.

H Drawings

H.1 On Tool




34
5
25
4
3
2
21,22
26,22
7
9
13
57
121
91
33
80
30
90
29,63
R 12
R 13
6,64
6,64
58,63
DRAWN
CHECKED
ENG APPR
MGR APPR
NAME
DATE
Mathias
01/06/16
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MILLIMETERS
ANGLES ±X.X°
2 PL ±X.XX 3 PL ±X.XXX
TITLE
Solid Edge
SIZE
A2
DWG NO
REV
FILE NAME: final_version_on_tool_adapter.dft
SCALE:
WEIGHT:
SHEET 1 OF 1
REVISION HISTORY
REV
DESCRIPTION
DATE
APPROVED

H.2 Test Rig

I Code

I.1 Test Rig

```
/*
* PlatformLedboard.ino version 2.0
* Author: Elin MK Nordmark
* Modified by: Bj rn Magnusson, Christoffer Lindahl
* Last modified: 2015-12-15
* This program lits the leds on the test rig when the
* corresponding platform sensors and/or
* test rig sensors are True.
*/


////////////////////////////////////////////////////////////
// This code lits up leds corresponding to the sensors on the
// test rig or the platform (sensor fusion).
// test rig sends by serial a BYTE corresponding to the ASCII
// of the letter of the bolt. It is read as a char.
// It is programmed onto an Arduino Mega 2560.
////////////////////////////////////////////////////////////

// X_t - the tool is on bolt X accordin to the sensors on the
     test rig
#define A_t 34
#define B_t 32
#define C_t 31
#define D_t 37
#define H_t 47
#define I_t 53
#define M_t 52
#define Q_t 50

// X_p - the tool is on bolt X according to the platfrom (
    sensor fusion of other systems).
#define A_p 33
#define B_p 35
#define C_p 39
#define D_p 41
#define H_p 43
#define I_p 45
#define M_p 49
#define Q_p 51

// X_in the inpin for bolt X test rig sensor. True when set
    LOW.
#define A_in 2
```

```
#define B_in 3
#define C_in 4
#define D_in 5
#define H_in 6
#define I_in 7
#define M_in 8
#define Q_in 9

void setup()
{
    pinMode(A_t, OUTPUT);
    pinMode(B_t, OUTPUT);
    pinMode(C_t, OUTPUT);
    pinMode(D_t, OUTPUT);
    pinMode(H_t, OUTPUT);
    pinMode(I_t, OUTPUT);
    pinMode(M_t, OUTPUT);
    pinMode(Q_t, OUTPUT);
//Second row/column:
    pinMode(A_p, OUTPUT);
    pinMode(B_p, OUTPUT);
    pinMode(C_p, OUTPUT);
    pinMode(D_p, OUTPUT);
    pinMode(H_p, OUTPUT);
    pinMode(I_p, OUTPUT);
    pinMode(M_p, OUTPUT);
    pinMode(Q_p, OUTPUT);

// Input pins
    pinMode(A_in, INPUT);
    pinMode(B_in, INPUT);
    pinMode(C_in, INPUT);
    pinMode(D_in, INPUT);
    pinMode(H_in, INPUT);
    pinMode(I_in, INPUT);
    pinMode(M_in, INPUT);
    pinMode(Q_in, INPUT);

// Pullup on input pins
    digitalWrite(A_in, HIGH);
    digitalWrite(B_in, HIGH);
    digitalWrite(C_in, HIGH);
    digitalWrite(D_in, HIGH);
    digitalWrite(H_in, HIGH);
    digitalWrite(I_in, HIGH);
    digitalWrite(M_in, HIGH);
    digitalWrite(Q_in, HIGH);
```

```
    Serial.begin(9600); // opens serial port to receive "in",
        sets data rate to 9600 bps
}// end setup()

void loop()
    {
// Invalue given from platform corresponding to bolt A,B,C,D,
   H,I,M or Q (ASCII number).
    static uint8_t in;

//Lit coresponding LED if on test rig sensor is TRUE,
   otherwise turn off LED.
    if  (digitalRead(A_in)==0)
        {
        digitalWrite(A_t, HIGH);
        Serial.write('A');
        }
    else
        {
        digitalWrite(A_t, LOW);
        }
    if  (digitalRead(B_in)==0)
        {
        digitalWrite(B_t, HIGH);
        Serial.write('B');
        }
    else
        {
        digitalWrite(B_t,LOW);
        }
    if  (digitalRead(C_in)==0)
        {
        digitalWrite(C_t, HIGH);
        Serial.write('C');
        }
    else
        {
        digitalWrite(C_t,LOW);
        }
    if  (digitalRead(D_in)==0)
        {
        digitalWrite(D_t, HIGH);
        Serial.write('D');
        }
    else
        {
        digitalWrite(D_t,LOW);
        }
    if  (digitalRead(H_in)==0)
```

```
        {
        digitalWrite(H_t, HIGH);
        Serial.write('H');
        }
    else
        {
        digitalWrite(H_t,LOW);
        }
    if   (digitalRead(I_in)==0)
        {
        digitalWrite(I_t, HIGH);
        Serial.write('I');
        }
    else
        {
        digitalWrite(I_t,LOW);
        }
    if   (digitalRead(M_in)==0)
        {
        digitalWrite(M_t, HIGH);
        Serial.write('M');
        }
    else
        {
        digitalWrite(M_t,LOW);
        }
    if   (digitalRead(Q_in)==0)
        {
        digitalWrite(Q_t, HIGH);
        Serial.write('Q');
        }
    else
        {
        digitalWrite(Q_t,LOW);
        }

// Read if platform sends
    while (Serial.available()>0)
        {
        in = Serial.read();

        //turn off LEDs when no accepted message is received
        if   ((in !='A') && (in !='B') && (in !='C') && (in
            !='D') && (in !='H') && (in !='I') && (in !='M')
            && (in !='Q'))
            {
            //Serial.write("error");
            digitalWrite(A_p, LOW);
            digitalWrite(B_p, LOW);
```

```
            digitalWrite(C_p, LOW);
            digitalWrite(D_p, LOW);
            digitalWrite(H_p, LOW);
            digitalWrite(I_p, LOW);
            digitalWrite(M_p, LOW);
            digitalWrite(Q_p, LOW);
            }
        //Lit corresponding LED if accepted message.
        if   (in == 'A')
            {
            digitalWrite(A_p, HIGH);
            }
        if   (in == 'B')
            {
            digitalWrite(B_p, HIGH);
            }
        if   (in == 'C')
            {
            digitalWrite(C_p, HIGH);
            }
        if   (in == 'D')
            {
            digitalWrite(D_p, HIGH);
            }
        if   (in == 'H')
            {
            digitalWrite(H_p, HIGH);
            }
        if   (in == 'I')
            {
            digitalWrite(I_p, HIGH);
            }
        if   (in == 'M')
            {
            digitalWrite(M_p, HIGH);
            }
        if   (in == 'Q')
            {
            digitalWrite(Q_p, HIGH);
            }
        }//end while

}// end loop()
```

I.2 Platform

I.2.1 Architecture

Main

```
#include <QCoreApplication>
#include <QObject>
#include <deque>
#include <queue>
#include <iostream>
#include <thread>          // Threading
#include <algorithm>       // do not know?
#include <mutex>           // To be thread safe
#include <sys/time.h>
#include "data.h"
#include "sv_receive.h"    // General setup of UDP server
#include <unistd.h>
#include "uwb_receive.h"
#include "Kalman.h"
#include "imu_receive.h"
#include "testrig.h"
#include <math.h>
#include <stdio.h>


Data testdata;



Kalman kalman_filter(&testdata);
IMU imu(&testdata);                        // Initalize the
    serial over USB
Testrig testrig(&testdata);                // Initalize the
    serial over USB

const int my_to_ms = 1000;                 // Turn microseconds
    to ms



void kalman_thread()
{
    std::cout << "Kalman thread Working \n";
    while (1){
        kalman_filter.applyKalman();
        usleep(20*my_to_ms);
    }
}

void imu_thread()
{
    std::cout << "Serial called\n";
    imu.imu_receive();
```

```
}

void testrig_thread()
{
    std::cout << "Serial called testrig\n";
    while(1)
    {
        testrig.testrig_receive();

    }
    std::cout << "Testrig stopped running" << std::endl;
}

void dist_thread()
{
    std::cout << "Distance called\n";
    while(1)
    {
        testdata.all_distance();
        usleep(10*my_to_ms);
    }
}

void populera_mera_tread()
{
    std::cout << "Populate called\n";
    while(1)
    {
        testdata.output.posX.addData(testdata.uWB.posX.
           getData(testdata.uWB.posX.newest));
        testdata.output.posY.addData(testdata.uWB.posY.
           getData(testdata.uWB.posY.newest));
        testdata.output.posZ.addData(testdata.uWB.posZ.
           getData(testdata.uWB.posZ.newest));

        usleep(10*my_to_ms);
    }
}

void LED_thread()
{
    std::cout << "Light LED\n";
    int i, min_i;
    double min;
    char letters [8] = {'A','B','C','D','I','H','M','Q'};
    while(1)
    {
        min_i = 0;
        min = testdata.dist[0];
```

```
        for (i=1;i<8;i++)
        {
            if (testdata.dist[i] < min) {min = testdata.dist[
                i]; min_i = i;}
        }
        if (min < 0.05)
           {
                testdata.bolt.near = letters[min_i];
                if(testdata.bolt.old != testdata.bolt.near)
                {
                        testdata.timeStart = testdata.now_int
                            ();
                        testdata.bolt.old = testdata.bolt.
                            near;
                }
            }
           else
            {
                testdata.bolt.near = '\0';
                testdata.bolt.old = '\0';
                testdata.timeStart = 0;
                testdata.bolt.saved = 0;
        }

        usleep(10*my_to_ms);
    }
}


int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    testdata.output.posX.addData(0);
    testdata.output.posY.addData(0);
    testdata.output.posZ.addData(0);
    testdata.uWB.posX.addData(0);
    testdata.uWB.posY.addData(0);
    testdata.uWB.posZ.addData(0);


    MyUDP svServer(&testdata);
    UWB_MyUDP uwbServer(&testdata);

    std::thread t4(imu_thread);
    std::thread t5(testrig_thread);
    std::thread t6(dist_thread);
    //std::thread t7(populera_mera_tread);
    std::thread t8(LED_thread);
    std::thread t3(kalman_thread);
```

```
    t3.detach();
    t4.detach();
    t5.detach();
    t6.detach();
    //t7.detach();
    t8.detach();

    std::cout << "Done with threads" << std::endl;

    return a.exec();

}
```

Data Class

```
#include "data.h"

Data::Data()
{

}

std::string Data::now_str()
{

    const boost::posix_time::ptime now =
        boost::posix_time::microsec_clock::local_time();
    const boost::posix_time::time_duration td = now.
       time_of_day();

    const long hours        = td.hours();
    const long minutes      = td.minutes();
    const long seconds      = td.seconds();
    const long milliseconds = td.total_milliseconds() -
                              ((hours * 3600 + minutes * 60 +
                                  seconds) * 1000);

    char buf[40];
    sprintf(buf, "%02ld:%02ld:%02ld.%03ld",
        hours, minutes, seconds, milliseconds);

    return buf;
}

long Data::now_int()
{
        const boost::posix_time::ptime now =
                boost::posix_time::microsec_clock::local_time
                   ();
```

```
        const boost::posix_time::time_duration td = now.
            time_of_day();

        const long milliseconds = td.total_milliseconds();

        return milliseconds;
}

void Data::saveToFile(std::string filename,double x,double y,
    double z,double xr,double yr,double zr)
{

        std::ofstream the_file(filename, std::ios::app);
                the_file << "T:" << now_str() << "," << x <<
                    "," << y << "," << z << "," << xr << ","
                                << yr << "," << zr << std::
                                    endl;
                the_file.close();
}

void Data::all_distance ()
{
        double  A_bolt[3]={0.20,0.175,0.015},
                B_bolt[3]={0.2,0.3,0.015},
                C_bolt[3]={0.1375,0.234,0.015},
                D_bolt[3]={0.2625,0.2375,0.015},
                I_bolt[3]={0.4,0.22,0.004},
                H_bolt[3]={0.32,0.7,0.004},
                M_bolt[3]={0.1,-0.027,-0.102},
                Q_bolt[3]={0.3,-0.027,-0.182};

        double now[3];

        now[0] = output.posX.getData(output.posX.newest);
        now[1] = output.posY.getData(output.posY.newest);
        now[2] = output.posZ.getData(output.posZ.newest);

        dist[0] = Data::distance(A_bolt, now);
        dist[1] = Data::distance(B_bolt, now);
        dist[2] = Data::distance(C_bolt, now);
        dist[3] = Data::distance(D_bolt, now);
        dist[4] = Data::distance(I_bolt, now);
        dist[5] = Data::distance(H_bolt, now);
        dist[6] = Data::distance(M_bolt, now);
        dist[7] = Data::distance(Q_bolt, now);


}
```

```
double Data::distance(double vector_a[3], double vector_b[3])
{
        double x, y, z,d;
        x = vector_a[0] - vector_b[0];
        y = vector_a[1] - vector_b[1];
        z = vector_a[2] - vector_b[2];
        d = pow((x*x+y*y+z*z),0.5);
        return d;
}

#ifndef DATA_H
#define DATA_H

#include "dataqueue.h"
#include <time.h>
#include <boost/date_time/posix_time/posix_time.hpp>
#include <fstream>
#include <vector>
#include <algorithm>
#include <stdexcept>


class Data
{

    public:
        Data();
        std::string now_str();
        long now_int();
        void all_distance();
        void saveToFile(std::string filename,double x, double
            y, double z,double xr = 0, double yr = 0, double
           zr = 0);
        struct IMU
        {
            DataQueue accX;
            DataQueue accY;
            DataQueue accZ;

            DataQueue quatW;
            DataQueue quatX;
            DataQueue quatY;
            DataQueue quatZ;
        };
        IMU iMU;

        struct UWB
        {
            DataQueue posX;
```

```
        DataQueue posY;
        DataQueue posZ;
    };
    UWB uWB;

    struct SV
    {
        DataQueue posX;
        DataQueue posY;
        DataQueue posZ;

        DataQueue quatW;
        DataQueue quatX;
        DataQueue quatY;
        DataQueue quatZ;


        DataQueue angX;
        DataQueue angY;
        DataQueue angZ;

        double frameno;
        int LOS;
    };
    SV sV;

    struct Output
    {
        DataQueue posX;
        DataQueue posY;
        DataQueue posZ;

        DataQueue quatW;
        DataQueue quatX;
        DataQueue quatY;
        DataQueue quatZ;

        DataQueue angX;
        DataQueue angY;
        DataQueue angZ;

        DataQueue new_update;

    };
    Output output;

    double dist[8] = {};

    struct Bolt
```

```
        {
            char on;
            char near = '\0';
            char old = '\0';
            char oldold = '\n';
            char latest = 'X';
            int saved = 0;
            int Asaved = 0;
            int Hsaved = 0;
            int Qsaved = 0;
        };
        Bolt bolt;

        int timeStart;
    private:
        double distance(double vector_a[3],double vector_b
            [3]);
        Data* testdata;


};
#endif // DATA_H

#ifndef DATAQUEUE_H
#define DATAQUEUE_H

#include <list>
#include <iostream>

class DataQueue
{
    public:
        DataQueue();
        double data[1000];
        int newest = 0;

        void tempf()
        {
            std::cout << "Testing tempf" << std::endl;
        }
        void addData (double value)
        {
            newest++;
            if(newest >= 1000) newest = 0;
            data[newest] = value;
        }

        double getData (int i)
        {
```

```
            return data[i];
        }

};

#endif // DATAQUEUE_H
```

IMU Functions

```
#include "imu_receive.h"
#include "Quaternion.h"



IMU::IMU(Data* testdata):  testdata(testdata)
{
        std::cout << "Default constructor for IMU" << std::
            endl;
        //std::cout << "T:" << testdata -> now_str() << '\n';
        std::ofstream the_file;

    /* My Arduino is on /dev/ttyUSB0 */
    char const *portname = "/dev/ttyUSB0";

    /* Open the file descriptor in non-blocking mode */
     fd = open(portname, O_RDWR | O_NOCTTY);

    /* Set up the control structure */
     struct termios toptions;

     /* Get currently set options for the tty */
     tcgetattr(fd, &toptions);

    /* Set custom options */

    /* 38400 baud */
     cfsetispeed(&toptions, B38400);
     cfsetospeed(&toptions, B38400);
     /* 8 bits, no parity, no stop bits */
     toptions.c_cflag &= ~PARENB;
     toptions.c_cflag &= ~CSTOPB;
     toptions.c_cflag &= ~CSIZE;
     toptions.c_cflag |= CS8;
     /* no hardware flow control */
     toptions.c_cflag &= ~CRTSCTS;
     /* enable receiver, ignore status lines */
     toptions.c_cflag |= CREAD | CLOCAL;
     /* disable input/output flow control, disable restart
        chars */
     toptions.c_iflag &= ~(IXON | IXOFF | IXANY);
```

```
     /* disable canonical input, disable echo,
     disable visually erase chars,
     disable terminal-generated signals */
     toptions.c_iflag &= ~(ICANON | ECHO | ECHOE | ISIG);
     /* disable output processing */
     toptions.c_oflag &= ~OPOST;
    /* no fixed amount of characters to come in before read
       returns */
     toptions.c_cc[VMIN] = 0;
     /* minimum time to wait before read returns */
     toptions.c_cc[VTIME] = 20;

    /* commit the options */
     tcsetattr(fd, TCSANOW, &toptions);

    /* Wait for the Arduino to reset */
     usleep(1000*1000);

     gyro_correction.giveValuesSV(0,M_PI,0);
}

int IMU::serialport_read_until(int fd, char* buf, char until)
{
    char b[1];
    int i=0;
    do {
        int n = read(fd, b, 1);  // read a char at a time
        if( n==-1) return -1;    // couldn't read
        if( n==0 ) {
            usleep( 10 * 1000 ); // wait 10 msec try again
            continue;
        }
        buf[i] = b[0]; i++;
    } while( b[0] != until );

    buf[i] = '\n';  // null terminate the string
    return 0;
}

void IMU::imu_receive(void)
{
        std::cout << "IMU started running" << std::endl;
        while(1)
        {
                // Flush anything already in the serial
                    buffer
            tcflush(fd, TCIFLUSH);

            error = serialport_read_until(fd, buf, '\n');
```

```
//clean the partbuffers so they don't contain
   values that might influence next reading

if(error == 0)
{

        for(i=0;i<10;i++){
             chaccX[i]=chaccY[i]=chaccZ[i]=
             chgyrX[i]=chgyrY[i]=chgyrZ[i]=
             chmX[i]=chmY[i]=chmZ[i] = '\0';
        }

        count = 1; i = 0;
        while(buf[i] != '\n'){
            if(buf[i] == ';') {i++; n = 0; count
               ++;}       //to skip the ; between
               the values
            if(count == 1)
            {chaccX[n]=buf[i]; n++;}
            else if(count == 2)
            {chaccY[n]=buf[i]; n++;}
            else if(count == 3)
            {chaccZ[n]=buf[i]; n++;}
            else if(count == 4)
            {chgyrX[n]=buf[i]; n++;}
            else if(count == 5)
            {chgyrY[n]=buf[i]; n++;}
            else if(count == 6)
            {chgyrZ[n]=buf[i]; n++;}
            else if(count == 7)
            {chmX[n]=buf[i]; n++;}
            else if(count == 8)
            {chmY[n]=buf[i]; n++;}
            else if(count == 9)
            {chmZ[n]=buf[i]; n++;}

            i++;
        }
        x = atof(chaccX);
        y = atof(chaccY);
        z = atof(chaccZ);
        xr = (atof(chgyrX)/131 + 1.2f);
        yr = (atof(chgyrY)/131 - 0.7f);
        zr = (atof(chgyrZ)/131 + 1.0f);

        /* ---------------------------------*/
        //        Popullate que here
```

```
                        if(x != 0){
                            gyro_now.giveValues(xr, yr, zr);
                            gyro_total = gyro_now*gyro_total;

                            testdata->iMU.accX.addData(x);
                            testdata->iMU.accY.addData(y);
                            testdata->iMU.accZ.addData(z);
                            testdata->iMU.quatW.addData(
                                gyro_total.w);
                            testdata->iMU.quatX.addData(
                                gyro_total.x);
                            testdata->iMU.quatY.addData(
                                gyro_total.y);
                            testdata->iMU.quatZ.addData(
                                gyro_total.z);
                        }
                        testdata->saveToFile("./Data/IMU_log.
                            csv",x,y,z,xr,yr,zr);
                }
        }
}

#ifndef IMU_H
#define IMU_H

#include <stdio.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#include <termios.h>
#include "data.h"
#include "Quaternion.h"

class IMU{
public:
    IMU(Data* testdata);
    void imu_receive();

private:

    int serialport_read_until(int fd, char* buf, char until);
    char chaccX[10],chaccY[10],chaccZ[10],chgyrX[10],chgyrY
        [10],chgyrZ[10],chmX[10],chmY[10],chmZ[10];
    double x;
    double y;
    double z;
    double xr;
    double yr;
    double zr;
    int i = 0,count = 0,n = 0,error = 0;
```

```
    char buf[256];
    int fd;
    Data* testdata;

    quaternion gyro_total;
    quaternion gyro_now;
    quaternion gyro_correction;
    StateVector rotated_gravityGY;
};

#endif // of IMU_H
```

SV Recieve Functions

```
#include "sv_receive.h"
#include "data.h"


MyUDP::MyUDP(Data* testdata):  testdata(testdata)
{
  std::cout << "creating sv object" << std::endl;
  socket = new QUdpSocket(this);
  host  = new QHostAddress("255.255.255.255");
  socket-> bind(*host,51001); //bind the socket to specific
      address and port
  connect(socket,SIGNAL(readyRead()),this,SLOT(readyRead()));
        //connect our signal and slot
}

void MyUDP::readyRead()
{

  QByteArray Buffer;//Data comes in, read
  Buffer.resize(socket->pendingDatagramSize());

  QHostAddress sender;   //who it is from and where it is
      from
  quint16 senderPort;
  socket->readDatagram(Buffer.data(),Buffer.size(),&sender,&
      senderPort);  //collect all the information

//     make framenumber readable;
  double fr;
  char bytes6[]={Buffer[0],Buffer[1],Buffer[2],Buffer[3]};
  memcpy(&fr, bytes6, sizeof(bytes6) );


//     make itemsInBlock readable;
  double it;
  char bytes7[]={Buffer[4]};
```

```
  memcpy(&it, bytes7, sizeof(bytes7) );


//     make itemID readable;
  double ID;
  char bytes8[]={Buffer[5]};
  memcpy(&ID, bytes8, sizeof(bytes8) );


//     make itemDataSize readable;
  double Da;
  char bytes9[]={Buffer[6],Buffer[7]};
  memcpy(&Da, bytes9, sizeof(bytes9) );


//     make xpos readable;

  char bytes0[]={Buffer[32],Buffer[33],Buffer[34],Buffer[35],
     Buffer[36],Buffer[37],Buffer[38],Buffer[39]};
  memcpy(&x, bytes0, sizeof(bytes0) );

//     make ypos readable;

  char bytes1[]={Buffer[40],Buffer[41],Buffer[42],Buffer[43],
     Buffer[44],Buffer[45],Buffer[46],Buffer[47]};
  memcpy(&y, bytes1, sizeof(bytes1) );

//     make zpos readable;

  char bytes2[]={Buffer[48],Buffer[49],Buffer[50],Buffer[51],
     Buffer[52],Buffer[53],Buffer[54],Buffer[55]};
  memcpy(&z, bytes2, sizeof(bytes2) );

//     make xrot readable;

  char bytes3[]={Buffer[56],Buffer[57],Buffer[58],Buffer[59],
     Buffer[60],Buffer[61],Buffer[62],Buffer[63]};
  memcpy(&xr, bytes3, sizeof(bytes3) );

//     make yrot readable;

  char bytes4[]={Buffer[64],Buffer[65],Buffer[66],Buffer[67],
     Buffer[68],Buffer[69],Buffer[70],Buffer[71]};
  memcpy(&yr, bytes4, sizeof(bytes4) );

//     make zrot readable;

  char bytes5[]={Buffer[72],Buffer[73],Buffer[74],Buffer[75],
     Buffer[76],Buffer[77],Buffer[78],Buffer[79]};
```

```
  memcpy(&zr, bytes5, sizeof(bytes5) );

  /* --------------------------------*/
  //        Popullate que here

  sv_vector.giveValues(0.182, 0 , -0.09);
  temp.x = x/1000.0f;
  temp.y = y/1000.0f;
  temp.z = z/1000.0f;
  sv_orientation.giveValuesSV(xr, yr, zr);

  sv_vector_rotated = sv_orientation.rotateVector(sv_vector);

  temp.x += sv_vector_rotated.x;
  temp.y -= sv_vector_rotated.y;
  temp.z += sv_vector_rotated.z;

  testdata->sV.frameno = fr;
  testdata->sV.posX.addData(x/1000.0f);
  testdata->sV.posY.addData(y/1000.0f);
  testdata->sV.posZ.addData(z/1000.0f);
  testdata->sV.angX.addData(xr);
  testdata->sV.angY.addData(yr);
  testdata->sV.angZ.addData(zr);

  testdata->sV.quatW.addData(sv_orientation.w);
  testdata->sV.quatX.addData(sv_orientation.x);
  testdata->sV.quatY.addData(sv_orientation.y);
  testdata->sV.quatZ.addData(sv_orientation.z);

  /* --------------------------------*/
  //        Save que here
  testdata->saveToFile("./Data/SV_log.csv",x,y,z,xr,yr,zr);
}

#ifndef MYUDP_H
#define MYUDP_H

#include <QObject>
#include <QDateTime>
#include <QFile>
#include <QUdpSocket>
#include <QTextStream>
#include <QHostAddress>
#include <stdio.h>
#include "data.h"
#include "Quaternion.h"
```

```
class MyUDP : public QObject
{
    Q_OBJECT
public:

    MyUDP(Data* testdata);

public slots:
    void readyRead();

private:
    QUdpSocket *socket;
    QHostAddress *host;
    QHostAddress *bcast;

    double x;
    double y;
    double z;
    double xr;
    double yr;
    double zr;
    Data* testdata;


    StateVector rotated_gravitySV;
    quaternion sv_orientation;
    StateVector sv_vector;
    StateVector temp;
    StateVector sv_vector_rotated;
};

#endif // MYUDP_H
```

UWB Recieve Functions

```
#include "uwb_receive.h"
#include "data.h"


UWB_MyUDP::UWB_MyUDP(Data* testdata): testdata(testdata)
{
    socket = new QUdpSocket(this);
    host   = new QHostAddress("192.168.10.2");
    socket-> bind(*host,51002); //bind the socket to specific
         address and port

    uwb_x_offset = -0.28;
    uwb_y_offset = 0.51;
    uwb_z_offset = -0.20;
```

```
    connect(socket,SIGNAL(readyRead()),this,SLOT(readyRead())
        );  //connect our signal and slot
}


void UWB_MyUDP::readyRead()
{
    char Buffer[24];
    int Buffer_t = 24;

    QHostAddress sender;   //who it is from and where it is
        from
    quint16 senderPort;
    socket->readDatagram(Buffer,Buffer_t,&sender,&senderPort)
        ;  //collect all the information

//     make xpos readable;

    char bytes0[]={Buffer[0],Buffer[1],Buffer[2],Buffer[3],
        Buffer[4],Buffer[5],Buffer[6],Buffer[7]};
    memcpy(&x, bytes0, sizeof(double) );

//     make ypos readable;

    char bytes1[]={Buffer[8],Buffer[9],Buffer[10],Buffer[11],
        Buffer[12],Buffer[13],Buffer[14],Buffer[15]};
    memcpy(&y, bytes1, sizeof(bytes1) );

//     make zpos readable;

    char bytes2[]={Buffer[16],Buffer[17],Buffer[18],Buffer
        [19],Buffer[20],Buffer[21],Buffer[22],Buffer[23]};
    memcpy(&z, bytes2, sizeof(bytes2) );

    testdata->uWB.posX.addData(x-uwb_x_offset);
    testdata->uWB.posY.addData(y-uwb_y_offset);
    testdata->uWB.posZ.addData(z-uwb_z_offset);


    if (testdata->sV.frameno != oldFrameno )
        {

        testdata->sV.LOS = 1;
        oldFrameno = testdata->sV.frameno;
        }
    else testdata->sV.LOS = 0;

    testdata->saveToFile("./Data/UWB_log.csv",x,y,z);
```

```
}

#ifndef UWB_MYUDP_H
#define UWB_MYUDP_H

#include <QObject>
#include <QDateTime>
#include <QUdpSocket>
#include <QTextStream>
#include <QHostAddress>
#include <chrono>
#include <stdio.h>
#include <string.h>
#include <QFile>
#include <fstream>
#include "data.h"

class UWB_MyUDP : public QObject
{
    Q_OBJECT
public:

    UWB_MyUDP(Data* testdata);

public slots:
    void readyRead();

private:
    QUdpSocket *socket;
    QHostAddress *host;
    QHostAddress *bcast;

    double x;
    double y;
    double z;

    double uwb_x_offset;
    double uwb_y_offset;
    double uwb_z_offset;
    double oldFrameno;
    QFile *the_file;
    QDateTime now ;
    Data* testdata;

};

#endif // MYUDP_H
```

Testrig Functions

```
#include "testrig.h"

Testrig::Testrig(Data* testdata) : testdata(testdata)
{

        std::cout << "Default constructor for Testrig" << std
           ::endl;
        std::cout << "T:" << testdata -> now_str() << '\n';

    /* My Arduino is on /dev/ttyACM1 can vary to ACM2 */
    char const *portnamerig = "/dev/ttyACM0";

    /* Open the file descriptor in non-blocking mode */
     fdrig = open(portnamerig, O_RDWR | O_NOCTTY);

    /* Set up the control structure */
     struct termios rigoptions;

     /* Get currently set options for the tty */
     tcgetattr(fdrig, &rigoptions);

    /* Set custom options */

    /* 38400 baud */
     cfsetispeed(&rigoptions, B38400);
     cfsetospeed(&rigoptions, B38400);
     /* 8 bits, no parity, no stop bits */
     rigoptions.c_cflag &= ~PARENB;
     rigoptions.c_cflag &= ~CSTOPB;
     rigoptions.c_cflag &= ~CSIZE;
     rigoptions.c_cflag |= CS8;
     /* no hardware flow control */
     rigoptions.c_cflag &= ~CRTSCTS;
     /* enable receiver, ignore status lines */
     rigoptions.c_cflag |= CREAD | CLOCAL;
     /* disable input/output flow control, disable restart
        chars */
     rigoptions.c_iflag &= ~(IXON | IXOFF | IXANY);
     /* disable canonical input, disable echo,
     disable visually erase chars,
     disable terminal-generated signals */
     rigoptions.c_iflag &= ~(ICANON | ECHO | ECHOE | ISIG);
     /* disable output processing */
     rigoptions.c_oflag &= ~OPOST;
     // copy comments from imu
     rigoptions.c_cc[VMIN] = 0;
     rigoptions.c_cc[VTIME] = 20;

    /* commit the options */
```

```
     tcsetattr(fdrig, TCSANOW, &rigoptions);

    /* Wait for the Arduino to reset */
     usleep(1000*1000);
}

int Testrig::serialport_writebyte( int fd, uint8_t b)
{
    int n = write(fd,&b,1);
    if( n!=1)
        return -1;
    return 0;
}


 int Testrig::serialport_read_until(int fd, char* buf, char
    until)
{
    char b[1];
    int i=0;
    do {
        int n = read(fd, b, 1);  // read a char at a time
        if( n==-1) return -1;    // couldn't read
        if( n==0 ) {
            usleep( 10 * 1000 ); // wait 10 msec try again
            continue;
        }
        buf[i] = b[0]; i++;

    } while( b[0] != until );

    buf[i] = '\0';  // null terminate the string
    return 0;
}

void Testrig::testrig_receive(void)
{

        tcflush(fdrig, TCIFLUSH);
    if(testdata->bolt.near != '\0')
    {
        serialport_writebyte(fdrig,testdata->bolt.near);
    }

        // Flush anything already in the serial buffer
    tcflush(fdrig, TCIFLUSH);

    error = serialport_read_until(fdrig, buf, '\n');
```

```
if(error == 0 && buf[0] != '\0' )
{
            testdata->bolt.on = buf[0];

    if((testdata->bolt.Asaved == 0 && testdata->bolt.on
       == 'A')
        || (testdata->bolt.Hsaved == 0 && testdata->bolt.
           on == 'H')
        || (testdata->bolt.Qsaved == 0 && testdata->bolt.
           on == 'Q'))
        {
        char add;
        if(testdata->bolt.on == 'Q') add = '\n';
        else add = ',';

        std::ofstream the_file("Data/Test_SF_data.csv",
           std::ios::app);
        the_file << buf[0]
        << "," << testdata->output.posX.getData(testdata
           ->output.posX.newest)
        << "," << testdata->output.posY.getData(testdata
           ->output.posY.newest)
        << "," << testdata->output.posZ.getData(testdata
           ->output.posZ.newest)
        << "," << testdata->output.angX.getData(testdata
           ->output.angX.newest)
        << "," << testdata->output.angY.getData(testdata
           ->output.angY.newest)
        << "," << testdata->output.angZ.getData(testdata
           ->output.angZ.newest)
        << "," << (testdata -> now_int()-testdata->
           timeStart)
        << add;
        the_file.close();

        std::ofstream File_SV("Data/Test_SV_data.csv",
           std::ios::app);
        File_SV << buf[0]
        << "," << testdata->sV.posX.getData(testdata->sV.
           posX.newest)
        << "," << testdata->sV.posY.getData(testdata->sV.
           posY.newest)
        << "," << testdata->sV.posZ.getData(testdata->sV.
           posZ.newest)
        << "," << testdata->sV.angX.getData(testdata->sV.
           angX.newest)
        << "," << testdata->sV.angY.getData(testdata->sV.
           angY.newest)
```

```
            << "," << testdata->sV.angZ.getData(testdata->sV.
                angZ.newest)
            << add;
            File_SV.close();

            std::ofstream File_UWB("Data/Test_UWB_data.csv",
                std::ios::app);
            File_UWB << buf[0]
            << "," << testdata->uWB.posX.getData(testdata->
                uWB.posX.newest)
            << "," << testdata->uWB.posY.getData(testdata->
                uWB.posY.newest)
            << "," << testdata->uWB.posZ.getData(testdata->
                uWB.posZ.newest)
            << add;
            File_UWB.close();

            if(testdata->bolt.on == 'A') {testdata->bolt.
                Asaved = 1; testdata->bolt.Hsaved = 0;
                testdata->bolt.Qsaved = 0; }
            if(testdata->bolt.on == 'H') {testdata->bolt.
                Asaved = 0; testdata->bolt.Hsaved = 1;
                testdata->bolt.Qsaved = 0; }
            if(testdata->bolt.on == 'Q') {testdata->bolt.
                Asaved = 0; testdata->bolt.Hsaved = 0;
                testdata->bolt.Qsaved = 1; }

            testdata->bolt.latest = buf[0];
            }
    }
}

#ifndef TESTRIG_H
#define TESTRIG_H

#include <stdio.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#include <termios.h>
#include "data.h"


class Testrig{
public:
    Testrig(Data* testdata);
    void testrig_receive();

private:
```

```
    int serialport_read_until(int fd, char* buf, char until);
    int serialport_writebyte( int fd, uint8_t b);
    int i = 0,count = 0,n = 0,error = 0;
    char buf[256];
    int fdrig;
    Data* testdata;

};

#endif // of TESTRIG
```

I.2.2 Sensor Fusion

```
#include "Kalman.h"


Kalman::Kalman(Data* testdata):testdata(testdata){

  delta_t        = 0.066;
  imu_variance   = 0.3;
  sv_variance    = 0.01*0.01;
  uwb_variance   = 0.15*0.15;
  process_noise = 0.0125;
  sv_vector.giveValues(0.182, 0 , -0.09);
  gravity.giveValues(0, 0, 9.81);
  uwb_vector.giveValues(0.21, 0, -0.135);

  std::cout<<"Default constructor Kalman\n";

  initial_covariance = 500;

    X.fill(0);
    //Random initial estimate of states
    X << 5.0 << arma::endr  //x
      << 0.0 << arma::endr  //x_dot
      << 0.0 << arma::endr  //x_dot_dot
      << 5.0 << arma::endr  //y
      << 0.0 << arma::endr  //y_dot
      << 0.0 << arma::endr  //y_dot_dot
      << 5.0 << arma::endr  //z
      << 0.0 << arma::endr  //z_dot
      << 0.0 << arma::endr; //z_dot_dot

    X.print("X: ");

    A << 1.0 << delta_t << 0.5*delta_t*delta_t << 0.0 << 0.0
       << 0.0 << 0.0 << 0.0 << 0.0 <<arma::endr
```

```
      << 0.0   << 1.0 << delta_t << 0.0 << 0.0 << 0.0 << 0.0
         << 0.0 << 0.0  <<arma::endr
      << 0.0   << 0.0 << 1.0 << 0.0 << 0.0 << 0.0 << 0.0 <<
         0.0 << 0.0 <<arma::endr
      << 0.0   << 0.0 << 0.0 << 1.0 << delta_t << 0.5*delta_t
         *delta_t << 0.0 << 0.0 << 0.0 <<arma::endr
      << 0.0   << 0.0 << 0.0 << 0.0 << 1.0 << delta_t << 0.0
         << 0.0 << 0.0  << arma::endr
      << 0.0   << 0.0 << 0.0 << 0.0 << 0.0 << 1.0 << 0.0 <<
         0.0 << 0.0  << arma::endr
      << 0.0   << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 1.0 <<
         delta_t << 0.5*delta_t*delta_t<<arma::endr
      << 0.0   << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 <<
         1.0 << delta_t<<arma::endr
      << 0.0   << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 <<
         0.0 << 1.0 <<arma::endr;

   A.print("A: ");



   Q = arma::zeros(num_states,num_states);

   Q(2,2) = process_noise;
   Q(5,5) = process_noise;
   Q(8,8) = process_noise;




   Q.print("Q: ");


   H << 1 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
       endr //x_uwb
      << 1 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //x_sv
      << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //x_dot_dot_imu
      << 0 << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //y_uwb
      << 0 << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //y_sv
      << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << 0 << arma::
         endr //y_dot_dot_imu
      << 0 << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << arma::
         endr //z_uwb
      << 0 << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << arma::
         endr //z_sv
```

```
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 1 << arma::
         endr; //z_dot_dot_imu

    H_los = H;

H_nlos << 1 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
   endr
      << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 1 << arma::
         endr;



    H.print("H: ");

    P = arma::eye(num_states,num_states)*initial_covariance;

    P.print("P: ");


   R  << uwb_variance << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0
      << arma::endr
      << 0 << sv_variance << 0 << 0 << 0 << 0 << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << imu_variance << 0 << 0 << 0 << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << uwb_variance << 0 << 0 << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << sv_variance << 0 << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << imu_variance << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << 0 << uwb_variance << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << sv_variance << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << imu_variance
         << arma::endr;

  R_los = R;
```

```
R_nlos<< uwb_variance << 0 << 0 << 0 << 0 << 0 << arma::endr
      << 0 << imu_variance << 0 << 0 << 0 << 0 << arma::endr
      << 0 << 0 << uwb_variance << 0 << 0 << 0 << arma::endr
      << 0 << 0 << 0 << imu_variance << 0 << 0 << arma::endr
      << 0 << 0 << 0 << 0 << uwb_variance << 0 << arma::endr
      << 0 << 0 << 0 << 0 << 0 << imu_variance << arma::endr;




    R.print("R: ");

    B.fill(0);

    B.print("B: ");

    u.fill(0);

    u.print("u: ");

    Z  << 0.0 <<arma::endr //x_uwb
       << 0.1 <<arma::endr //x_sv
       << 0.1 <<arma::endr //x_dot_dot_imu
       << 0.0 <<arma::endr //y_uwb
       << 0.1 <<arma::endr //y_sv
       << 0.1 <<arma::endr //y_dot_dot_imu
       << 0.0 <<arma::endr //z_uwb
       << 0.1 <<arma::endr //z_sv
       << 0.1 <<arma::endr;//z_dot_dot_imu

    Z.print("Z: ");

}

Kalman::Kalman(Data* test_data, double uwb_var, double
   imu_var, double sv_var, double initial_covar, double
   process_nois):
  testdata(test_data), uwb_variance(uwb_var), imu_variance(
     imu_var), sv_variance(sv_var),
  initial_covariance(initial_covar), process_noise(
     process_nois)
{

    sv_vector.giveValues(0.182, 0 , -0.09);
    gravity.giveValues(0, 0, 9.81);
    uwb_vector.giveValues(0.21, 0, -0.135);
    delta_t = 0.066;
```

```
X.fill(0);
//Random initial estimate of states
X << 5.0 << arma::endr  //x
  << 0.0 << arma::endr  //x_dot
  << 0.0 << arma::endr  //x_dot_dot
  << 5.0 << arma::endr  //y
  << 0.0 << arma::endr  //y_dot
  << 0.0 << arma::endr  //y_dot_dot
  << 5.0 << arma::endr  //z
  << 0.0 << arma::endr  //z_dot
  << 0.0 << arma::endr; //z_dot_dot

X.print("X: ");

A << 1.0 << delta_t << 0.5*delta_t*delta_t  << 0.0 << 0.0
    << 0.0 << 0.0 << 0.0 << 0.0 <<arma::endr
  << 0.0 << 1.0 << delta_t     << 0.0 << 0.0 << 0.0 << 0.0
      << 0.0 << 0.0   <<arma::endr
  << 0.0 << 0.0 << 1.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0
      << 0.0 <<arma::endr
  << 0.0 << 0.0 << 0.0 << 1.0 << delta_t << 0.5*delta_t*
     delta_t << 0.0 << 0.0 << 0.0 <<arma::endr
  << 0.0 << 0.0 << 0.0 << 0.0 << 1.0 << delta_t << 0.0 <<
      0.0 << 0.0  << arma::endr
  << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 1.0 << 0.0 << 0.0
      << 0.0  << arma::endr
  << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 1.0 <<
     delta_t << 0.5*delta_t*delta_t<<arma::endr
  << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 1.0
      << delta_t<<arma::endr
  << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0 << 0.0
      << 1.0 <<arma::endr;

A.print("A: ");



Q = arma::zeros(num_states,num_states);

Q(2,2) = process_noise;
Q(5,5) = process_noise;
Q(8,8) = process_noise;



Q.print("Q: ");
```

```
    H << 1 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
       endr //x_uwb
      << 1 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //x_sv
      << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //x_dot_dot_imu
      << 0 << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //y_uwb
      << 0 << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << arma::
         endr //y_sv
      << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << 0 << arma::
         endr //y_dot_dot_imu
      << 0 << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << arma::
         endr //z_uwb
      << 0 << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << arma::
         endr //z_sv
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 1 << arma::
         endr; //z_dot_dot_imu

    H_los = H;



H_nlos<< 1 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
    endr
      << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 1 << 0 << 0 << 0 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 1 << 0 << 0 << arma::
         endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 1 << arma::
         endr;

    H.print("H: ");

    P = arma::eye(num_states,num_states)*initial_covariance;

    P.print("P: ");


    R << uwb_variance << 0<< 0 << 0 << 0 << 0 << 0 << 0 << 0
       << arma::endr
      << 0 << sv_variance << 0 << 0 << 0 << 0 << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << imu_variance<< 0 << 0 << 0 << 0 << 0 << 0
         << arma::endr
```

```
      << 0 << 0 << 0 << uwb_variance<< 0 << 0 << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << sv_variance << 0 << 0 << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << imu_variance<< 0 << 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << 0 << uwb_variance<< 0 << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << sv_variance << 0
         << arma::endr
      << 0 << 0 << 0 << 0 << 0 << 0 << 0 << 0 << imu_variance
         << arma::endr;

    R_los = R;


R_nlos<< uwb_variance << 0 << 0 << 0 << 0 << 0 << arma::endr
      << 0 << imu_variance << 0 << 0 << 0 << 0 << arma::endr
      << 0 << 0 << uwb_variance << 0 << 0 << 0 << arma::endr
      << 0 << 0 << 0 << imu_variance << 0 << 0 << arma::endr
      << 0 << 0 << 0 << 0 << uwb_variance << 0 << arma::endr
      << 0 << 0 << 0 << 0 << 0 << imu_variance << arma::endr;


    R.print("R: ");

    B.fill(0);

    B.print("B: ");

    u.fill(0);

    u.print("u: ");

    Z << 0.0 <<arma::endr //x_uwb
      << 0.1 <<arma::endr //x_sv
      << 0.1 <<arma::endr //x_dot_dot_imu
      << 0.0 <<arma::endr //y_uwb
      << 0.1 <<arma::endr //y_sv
      << 0.1 <<arma::endr //y_dot_dot_imu
      << 0.0 <<arma::endr //z_uwb
      << 0.1 <<arma::endr //z_sv
      << 0.1 <<arma::endr;//z_dot_dot_imu

    Z.print("Z: ");


}
```

```
void Kalman::rotateStates(){
    //
       ////////////////////////////////////////////////////////////

    // removing gravity
    //
       ////////////////////////////////////////////////////////////


    rotated_gravity = q_average.getInverse().rotateVector(
       accelerometer);

    acc_x_real = rotated_gravity.x - 0;
    acc_y_real = rotated_gravity.y - 0;
    acc_z_real = rotated_gravity.z - 9.81f;

    //
       ////////////////////////////////////////////////////////////

    // adjusting SV position to the tip
    //
       ////////////////////////////////////////////////////////////


    sv_vector_rotated = q_average.rotateVector(sv_vector);

    real_pos_sv.x = testdata->sV.posX.getData(testdata->sV.
       posX.newest);
    real_pos_sv.y = testdata->sV.posY.getData(testdata->sV.
       posY.newest);
    real_pos_sv.z = testdata->sV.posZ.getData(testdata->sV.
       posZ.newest);

    real_pos_sv.x += sv_vector_rotated.x;
    real_pos_sv.y -= sv_vector_rotated.y;
    real_pos_sv.z += sv_vector_rotated.z;

    //
       ///////////////////////////////////////////////////////////

    //Adjusting UWB
    //
       ///////////////////////////////////////////////////////////

    uwb_vector_rotated = q_average.rotateVector(uwb_vector);
    real_pos_uwb.x = testdata->uWB.posX.getData(testdata->uWB
       .posX.newest);
    real_pos_uwb.y = testdata->uWB.posY.getData(testdata->uWB
       .posY.newest);
```

```
    real_pos_uwb.z = testdata->uWB.posZ.getData(testdata->uWB
       .posZ.newest);

    real_pos_uwb.x += uwb_vector_rotated.x;
    real_pos_uwb.y -= uwb_vector_rotated.y;
    real_pos_uwb.z += uwb_vector_rotated.z;

    //
       /////////////////////////////////////////////////////////


}


void Kalman::applyKalman()
{

    //
       //////////////////////////////////////////////////////////

    //Converting accelerometer values to meters/second
    //
       //////////////////////////////////////////////////////////

    accelerometer.x = 9.81f *testdata->iMU.accX.getData(
       testdata->iMU.accX.newest)/16457.0f;
    accelerometer.y = 9.81f *testdata->iMU.accY.getData(
       testdata->iMU.accY.newest)/16457.0f;
    accelerometer.z = 9.81f *testdata->iMU.accZ.getData(
       testdata->iMU.accZ.newest)/16457.0f;

   // std::cout <<"Acc: "<< accelerometer.x << ", "<<
      accelerometer.y << ", "<< accelerometer.z << std::endl;


    gyro_rot.w= testdata->iMU.quatW.getData(testdata->iMU.
       quatW.newest);
    gyro_rot.x= testdata->iMU.quatX.getData(testdata->iMU.
       quatX.newest);
    gyro_rot.y= testdata->iMU.quatY.getData(testdata->iMU.
       quatY.newest);
    gyro_rot.z= testdata->iMU.quatZ.getData(testdata->iMU.
       quatZ.newest);

    Quaternion_averager stuff;

    //correcting IMU drift;
```

```
if(testdata->sV.LOS)
{

  if(entered_los)
  {
      //If we just entered Los, resize
      std::cout<<"Entered Los Mode"<<std::endl;
      R.zeros(num_sensors,num_sensors);
      H.zeros(num_sensors,num_states);
      Z.zeros(num_sensors,1);
      R = R_los;
      H = H_los;

      entered_los = 0;


  }
  sv_rot.w = testdata->sV.quatW.getData(testdata->sV.quatW.
     newest);
  sv_rot.x = testdata->sV.quatX.getData(testdata->sV.quatX.
     newest);
  sv_rot.y = testdata->sV.quatY.getData(testdata->sV.quatY.
     newest);
  sv_rot.z = testdata->sV.quatZ.getData(testdata->sV.quatZ.
     newest);


    //If we were already in Los, just update the
       measurement
  std::cout<<"Los Mode"<<std::endl;


  if(isnan(gyro_rot.x))
  {
    std::cout<<"Gyro is Nan"<<std::endl;
    gyro_rot.w = 1.0;
    gyro_rot.x = 0.0;
    gyro_rot.y = 0.0;
    gyro_rot.z = 0.0;

  }

  if(isnan(rotated_gravity.x))
  {
    std::cout<<"Rotated gravity is Nan +++++++++++"<<std::
       endl;

    acc_x_real=0.0;
    acc_y_real=0.0;
```

```
    acc_z_real=0.0;


  }

  q_average = stuff.average(sv_rot, gyro_rot);




  //
     ////////////////////////////////////////////////////////////

  // removing gravity
  //
     ////////////////////////////////////////////////////////////


  rotated_gravity = q_average.getInverse().rotateVector(
     accelerometer);


  if(isnan(rotated_gravity.x))
  {
    std::cout<<"Rotated gravity is Nan ---------"<<std::
       endl;

    acc_x_real=0.0;
    acc_y_real=0.0;
    acc_z_real=0.0;


  }

  else{

  acc_x_real = rotated_gravity.x - 0.0f;
  acc_y_real = rotated_gravity.y - 0.0f;
  acc_z_real = rotated_gravity.z - 9.81f;
}

  //
     ////////////////////////////////////////////////////////////

  // adjusting SV position to the tip
  //
     ////////////////////////////////////////////////////////////
```

```
sv_vector_rotated = q_average.rotateVector(sv_vector);

real_pos_sv.x = testdata->sV.posX.getData(testdata->sV.
    posX.newest);
real_pos_sv.y = testdata->sV.posY.getData(testdata->sV.
    posY.newest);
real_pos_sv.z = testdata->sV.posZ.getData(testdata->sV.
    posZ.newest);

std::cout << "SV real pos = " <<  testdata->sV.posX.
    getData(testdata->sV.posX.newest) << ", " <<  testdata
    ->sV.posY.getData(testdata->sV.posY.newest) << ", "
<<  testdata->sV.posZ.getData(testdata->sV.posZ.newest)
    << std::endl;
std::cout << "SV rotated = " <<  sv_vector_rotated.x << "
    , " << sv_vector_rotated.y << ", " <<
    sv_vector_rotated.z << std::endl;

real_pos_sv.x += sv_vector_rotated.x;
real_pos_sv.y -= sv_vector_rotated.y;
real_pos_sv.z += sv_vector_rotated.z;


//
    ///////////////////////////////////////////////////////////

//Adjusting UWB
//
    ///////////////////////////////////////////////////////////

uwb_vector_rotated = q_average.rotateVector(uwb_vector);
real_pos_uwb.x = testdata->uWB.posX.getData(testdata->uWB
    .posX.newest);
real_pos_uwb.y = testdata->uWB.posY.getData(testdata->uWB
    .posY.newest);
real_pos_uwb.z = testdata->uWB.posZ.getData(testdata->uWB
    .posZ.newest);

real_pos_uwb.x += uwb_vector_rotated.x;
real_pos_uwb.y -= uwb_vector_rotated.y;
real_pos_uwb.z += uwb_vector_rotated.z;

//
    //////////////////////////////////////////////////////////


//
    //////////////////////////////////////////////////////////
```

```
    Z << real_pos_uwb.x <<arma::endr
      << real_pos_sv.x  <<arma::endr
      << acc_x_real     <<arma::endr
      << real_pos_uwb.y <<arma::endr
      << real_pos_sv.y  <<arma::endr
      << acc_y_real     <<arma::endr
      << real_pos_uwb.z <<arma::endr
      << real_pos_sv.z  <<arma::endr
      << acc_z_real     <<arma::endr;


  gyro_rot.x = q_average.x;
  gyro_rot.y = q_average.y;
  gyro_rot.z = q_average.z;

  entered_nlos = 1; //Setting the flag for next time we
     enter NLOS
}

if(!testdata->sV.LOS)
{
  //If we're in N-los
  if(entered_nlos)
  {


    last_known_x =testdata->sV.posX.getData(testdata->sV.
       posX.newest); //Using imu point
    last_known_y =testdata->sV.posY.getData(testdata->sV.
       posY.newest);
    last_known_z =testdata->sV.posZ.getData(testdata->sV.
       posZ.newest);

    uwb_error_x =testdata->uWB.posX.getData(testdata->uWB.
       posX.newest)  - last_known_x; //Calculating error in
        the tag point
    uwb_error_y =testdata->uWB.posY.getData(testdata->uWB.
       posY.newest)  - last_known_y;
    uwb_error_z =testdata->uWB.posZ.getData(testdata->uWB.
       posZ.newest)  - last_known_z;

    entered_nlos = 0; //Resetting entered flag

  }
```

```
std::cout<<"N-los Mode  "<<std::endl;
R.zeros(num_sensors_nlos,num_sensors_nlos);
H.zeros(num_sensors_nlos,num_states);
R = R_nlos;
H = H_nlos;


 if(isnan(gyro_rot.x))
{
  std::cout<<"Gyro is Nan \n";
  gyro_rot.w = 1.0;
  gyro_rot.x = 0.0;
  gyro_rot.y = 0.0;
  gyro_rot.z = 0.0;

}

q_average = gyro_rot;
/////////////////////////////////////////
// removing gravity
////////////////////////////////////////




///////////////////////////////////
//Adjusting UWB
//////////////////////////////////
uwb_vector_rotated = q_average.rotateVector(uwb_vector);
real_pos_uwb.x = testdata->uWB.posX.getData(testdata->uWB
   .posX.newest);
real_pos_uwb.y = testdata->uWB.posY.getData(testdata->uWB
   .posY.newest);
real_pos_uwb.z = testdata->uWB.posZ.getData(testdata->uWB
   .posZ.newest);

real_pos_uwb.x += uwb_vector_rotated.x;
real_pos_uwb.y -= uwb_vector_rotated.y;
real_pos_uwb.z += uwb_vector_rotated.z;





rotated_gravity = q_average.getInverse().rotateVector(
   accelerometer);
```

```
    if(isnan(rotated_gravity.x))
    {
      std::cout<<"Rotated gravity is Nan"<<std::endl;

      acc_x_real=0.0;
      acc_y_real=0.0;
      acc_z_real=0.0;


    }

    else{

    acc_x_real = rotated_gravity.x - 0.0f;
    acc_y_real = rotated_gravity.y - 0.0f;
    acc_z_real = rotated_gravity.z - 9.81f;
  }




   Z << real_pos_uwb.x - uwb_error_x <<arma::endr //Switching
        off imu so I can see the UWB clearer
     << acc_x_real      <<arma::endr
     << real_pos_uwb.y - uwb_error_y <<arma::endr
     << acc_y_real      <<arma::endr
     << real_pos_uwb.z - uwb_error_z <<arma::endr
     << acc_z_real      <<arma::endr;

     entered_los = 1; //Set flag for next time we enter Los;

     std::cout<<"UWB error: "<<uwb_error_x<<", "<<uwb_error_y
        <<", "<<uwb_error_z<<std::endl;
  }



//Filtering




    //Prediction
    X = A*X; //+ B*u;





    P = A*P*trans(A) + Q;
```

```
    //Compute Kalman gain
    K = P*trans(H)*inv(H*P*trans(H)+ R);

    //Correct based on observation
    X = X + K*(Z - H*X); //Estimate
    P = P - K*H*P;

    testdata->output.posX.addData(X(0));
    testdata->output.posY.addData(X(3));
    testdata->output.posZ.addData(X(6));
    testdata->saveToFile("./Data/output_log.csv",X(0),X(3),X
        (6));
    testdata->output.quatW.addData(q_average.w);
    testdata->output.quatX.addData(q_average.x);
    testdata->output.quatY.addData(q_average.y);
    testdata->output.quatZ.addData(q_average.z);
    testdata->output.new_update.addData(2.0f);


    std::cout<<"Estimated positions:"<<X(0)<<" ,"<<X(3)<<" ,"
        <<X(6)<<std::endl;
    std::cout<<"Estimated velocity:"<<X(1)<<" ,"<<X(4)<<" ,"
        <<X(7)<<std::endl;
    std::cout<<"Estimated accelerations:"<<X(2)<<" ,"<<X(5)<<
        " ,"<<X(8)<<std::endl<<std::endl<<std::endl;

    //X.print("X estimate: ");


}
```

I.3 Example Adaptive Kalman Filter

```
    //Prediction

    X = A*X; //+ B*u;




    P = A*P*trans(A) + Q;




    //Compute Kalman gain
    K = P*trans(H)*inv(H*P*trans(H)+ R);
```

```
    y = Z - H*X;

    //If Residual is large, increase process noise

    if(y>0.5)
    {
        Q = Q*100;
    }

    else
    {
        Q = Q_init;
    }

    //Correct based on observation

    X = X + K*(Z - H*X); //Estimate

    P = P - K*H*P;
```

I.4 On Tool

```
/* Connection pins:
Arduino      MARG MPU-9150
  A5             SCL
  A4             SDA
  3.3V           VCC
  GND            GND
  D2             INT
*/

int pin = 3;


#include "Wire.h"
#include "I2Cdev.h"
#include "MPU6050.h"

// class default I2C address is 0x68
// specific I2C addresses may be passed as a parameter here
// AD0 low = 0x68 (default for InvenSense evaluation board)
// AD0 high = 0x69
MPU6050 accelgyro;

int16_t ax, ay, az;
int16_t gx, gy, gz;
int16_t mx, my, mz;
```

```
#define LED_PIN 13
bool blinkState = false;

void setup() {

pinMode(pin, OUTPUT);

    // join I2C bus (I2Cdev library doesn't do this
       automatically)
    Wire.begin();

    // initialize serial communication
    // (38400 chosen because it works as well at 8MHz as it
       does at 16MHz, but
    // it's really up to you depending on your project)
    Serial.begin(38400);

    // initialize device
    Serial.println("Initializing IM2C devices...");
    accelgyro.initialize();

    // verify connection
    Serial.println("Testing device connections...");
    Serial.println(accelgyro.testConnection() ? "MPU6050
       connection successful" : "MPU6050 connection failed");

    // configure Arduino LED for
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    // read raw accel/gyro measurements from device
    accelgyro.getMotion9(&ax, &ay, &az, &gx, &gy, &gz, &mx, &
       my, &mz);

    // these methods (and a few others) are also available
    //accelgyro.getAcceleration(&ax, &ay, &az);
    //accelgyro.getRotation(&gx, &gy, &gz);

    // display tab-separated accel/gyro x/y/z values
    //Serial.println(pinState); //Serial.print("\t");

    Serial.print("a/g/m:\t");
    Serial.print(ax); //Serial.print("\t");
    Serial.print(";");
    Serial.print(ay); //Serial.print("\t");
    Serial.print(";");
    Serial.print(az); //Serial.print("\t");
```

```
    Serial.print(";");
    Serial.print(gx); //Serial.print("\t");
    Serial.print(";");
    Serial.print(gy); //Serial.print("\t");
    Serial.print(";");
    Serial.print(gz); //Serial.print("\t");
    Serial.print(";");
    Serial.print(mx); //Serial.print("\t");
    Serial.print(";");
    Serial.print(my); //Serial.print("\t");
    Serial.print(";");
    Serial.print(mz);
    Serial.print(";");
    Serial.println(digitalRead(pin));

    // blink LED to indicate activity
    blinkState = !blinkState;
    digitalWrite(LED_PIN, blinkState);

}
```

J Test data

J.1 Test data for the individual systems

This appendix shows the test data for the unmitigted systems. Important to consider here is that the Accuracy is measured as the euclidean distance from systems output positions and the actual bolt position.

Test	UWB Accuracy [m]			SV Accuracy [m]		
Nr	A	H	Q	A	H	Q
1	0,960	1,538	0,629	0,270	0,770	0,351
2	1,011	1,622	1,340	0,253	0,405	0,217
3	0,946	1,644	0,675	0,256	0,217	0,218
4	0,991	1,751	0,761	0,239	0,214	0,222
5	1,752	1,602	1,632	0,220	0,658	0,215
6	1,736	1,618	0,674	0,214	0,224	0,221
7	1,002	1,623	1,176	0,228	0,384	0,218
8	0,991	1,646	0,720	0,244	0,250	0,213
9	0,940	1,678	1,261	0,220	0,393	0,216
10	3,900	1,586	0,798	0,227	0,201	0,214
11	1,065	1,666	2,195	0,277	0,391	0,216
12	0,980	1,670	0,689	0,275	0,213	0,215
13	2,123	1,595	0,648	0,234	0,376	0,217
14	3,788	1,658	0,785	0,230	0,388	0,216
15	1,308	1,705	0,987	0,281	0,381	0,215
16	0,961	1,597	0,608	0,244	0,385	0,217
17	1,679	1,557	0,859	0,256	0,395	0,224
18	0,873	1,668	0,796	0,229	0,383	0,216
19	3,515	1,459	0,955	0,221	0,389	0,215
20	0,956	1,549	0,703	0,227	0,200	0,221
21	2,000	1,613	0,635	0,259	0,384	0,212
22	0,837	1,778	0,714	0,274	0,387	0,224
23	0,861	1,668	0,565	0,269	0,375	0,221
24	0,911	1,562	0,718	0,296	0,388	0,215
25	4,188	1,615	0,727	0,262	0,389	0,220

Test	UWB Accuracy [m]				SV Accuracy [m]		
Nr	A	H	Q		A	H	Q
26	1,188	1,763	0,771		0,280	0,283	0,215
27	2,798	1,769	1,048		0,225	0,310	0,218
28	1,660	1,681	0,808		0,244	0,215	0,226
29	0,797	1,535	0,614		0,291	0,189	0,221
30	0,954	1,513	0,564		0,267	0,383	0,216
31	0,996	1,495	2,066		0,230	0,364	0,216
32	1,381	1,592	0,624		0,233	0,391	0,217
33	3,437	1,430	0,621		0,223	0,399	0,224
34	1,006	1,511	1,072		0,245	0,399	0,214
35	1,004	1,694	0,641		0,263	0,404	0,216
36	1,440	1,675	0,991		0,257	0,390	0,217
37	1,052	1,509	0,534		0,306	0,387	0,217
38	3,646	1,456	0,586		0,282	0,408	0,222
39	1,012	2,263	0,586		0,238	0,373	0,223
40	1,015	1,478	0,980		0,225	0,382	0,215
41	0,994	1,547	0,771		0,249	0,381	0,217
42	1,026	1,580	0,573		0,232	0,384	0,218
43	0,926	1,504	1,197		0,253	0,379	0,218
44	3,544	1,529	0,641		0,228	0,385	0,214
45	4,184	1,438	0,722		0,223	0,383	0,214
46	2,532	1,512	1,380		0,237	0,389	0,223
47	0,981	1,438	1,353		0,265	0,382	0,216
48	1,001	1,521	1,182		0,250	0,393	0,224
49	2,889	1,436	0,634		0,200	0,393	0,220
50	0,934	1,477	0,584		0,232	0,398	0,214
51	1,046	1,520	1,363		0,254	0,207	0,212
52	0,994	1,595	0,753		0,219	0,195	0,217
53	0,931	1,714	1,086		0,247	0,400	0,214
54	0,985	1,680	0,633		0,253	0,210	0,216
55	0,956	1,628	0,638		0,237	0,208	0,213
56	0,818	1,498	1,417		0,222	0,208	0,218
57	0,924	1,742	0,722		0,248	0,289	0,218
58	0,794	1,605	0,767		0,243	0,212	0,218
59	0,936	1,532	0,664		0,298	0,213	0,215
60	1,048	1,550	0,728		0,279	0,387	0,220
61	0,858	1,608	1,533		0,264	0,269	0,215
62	0,853	1,603	0,594		0,266	0,213	0,216
63	0,872	1,639	0,669		0,293	0,277	0,220
64	0,842	1,648	0,590		0,283	0,205	0,219
65	0,924	1,698	0,623		0,255	0,388	0,216

Test	UWB Accuracy [m]				SV Accuracy [m]		
Nr	A	H	Q		A	H	Q
66	0,874	1,459	0,599		0,267	0,214	0,217
67	0,848	1,534	0,882		0,252	0,202	0,221
68	0,932	1,582	0,624		0,278	0,198	0,214
69	0,764	1,650	0,561		0,283	0,226	0,220
70	0,887	1,584	2,333		0,298	0,221	0,215
71	0,815	1,436	2,365		0,288	0,200	0,218
72	0,818	1,548	0,706		0,304	0,206	0,214
73	0,939	1,539	0,573		0,246	0,203	0,221
74	0,958	1,442	0,566		0,213	0,388	0,218
75	1,024	1,545	0,617		0,189	0,394	0,227
76	1,003	1,598	0,605		0,175	0,360	0,220
77	1,023	1,382	0,588		0,209	0,399	0,219
78	2,410	1,383	0,620		0,223	0,380	0,213
79	1,025	1,666	0,677		0,216	0,177	0,222
80	1,067	1,473	0,545		0,224	0,382	0,217
81	1,117	1,488	1,066		0,249	0,386	0,219
82	1,101	1,530	2,003		0,226	0,373	0,201
83	1,108	1,592	1,291		0,226	0,373	0,218
84	1,121	1,485	2,004		0,224	0,384	0,225
85	3,524	1,464	0,593		0,203	0,394	0,215
86	1,036	1,660	0,932		0,200	0,391	0,215
87	0,974	1,581	0,566		0,230	0,382	0,215
88	1,065	1,548	0,931		0,141	0,391	0,217
89	0,922	2,360	0,699		0,239	0,394	0,213
90	0,915	2,675	1,777		0,153	0,380	0,232
91	1,030	1,592	1,005		0,146	0,391	0,216
92	0,952	1,595	0,864		0,143	0,380	0,221
93	0,956	1,737	0,684		0,148	0,390	0,217
94	0,914	1,457	0,620		0,148	0,387	0,219
95	0,860	1,461	0,665		0,137	0,395	0,217
96	0,933	1,489	0,588		0,233	0,392	0,216
97	0,966	3,480	0,822		0,145	0,377	0,216
98	0,945	1,574	1,251		0,171	0,381	0,225
99	0,896	1,583	0,663		0,141	0,386	0,212
100	1,021	1,455	1,265		0,154	0,381	0,216
101	0,945	1,637	1,580		0,152	0,380	0,212
102	1,006	1,517	0,582		0,147	0,391	0,213
103	0,887	1,663	0,797		0,147	0,383	0,216
104	1,015	1,597	2,053		0,150	0,401	0,224

J.2 Test data for the final system

This appendix shows the test data for the final system. Important to consider here is that the Accuracy is measured as the euclidean distance from system output position and the actual bolt. The "Lag ok? [y/n]" column is the observed delay identified by the eye and y means that there was no delay noticed. The "time difference [ms]" column is the time difference = system time - touching the bolt. I.e. a positive number means that the system recognise the correct bolt even before touching it. If the number is very large (>100000) the system did not recognise the bolt or the system was recognising it after the touch.

Test	Accuracy [m]			lag ok? [y/n]			Time difference [ms]		
Nr	A	H	Q	A	H	Q	A	H	Q
1	0,036	0,502	0,098	y	n	n	155	35037035	35040443
2	0,032	0,808	0,087	y	n	n	232	35133390	35136467
3	0,024	0,048	0,082	y	y	n	149	5	35147123
4	0,044	0,645	0,087	y	n	n	35152706	35155044	35158391
5	0,045	0,050	0,060	y	y	n	35162925	35165513	35168647
6	0,039	0,176	0,075	y	y	n	17	35175908	35179066
7	0,042	0,124	0,067	y	n	n	178	35186558	35189801
8	0,038	0,249	0,054	y	y	y	106	35196977	35200549
9	0,042	0,094	0,075	y	n	n	33	35209314	35212382
10	0,032	0,132	0,061	y	n	n	129	35218353	35221106
11	0,031	0,038	0,078	y	y	n	92	343	35573756
12	0,018	0,556	0,063	y	n	n	354	35580797	35583369
13	0,031	0,972	0,075	y	n	n	365	35589996	35594063
14	0,027	0,460	0,048	y	n	y	215	35600535	35603955
15	0,025	0,373	0,047	y	n	y	213	35611016	35613359
16	0,033	0,213	0,058	y	n	y	176	35620481	35623557
17	0,052	0,483	0,066	n	n	n	35632122	35635067	35638458
18	0,028	0,304	0,051	y	n	n	434	35653735	35656725
19	0,024	0,373	0,046	y	n	y	353	35668292	1
20	0,023	0,311	0,067	y	n	n	126	35698335	35700731
21	0,045	0,429	0,062	y	n	n	458	35707682	35710422
22	0,038	0,462	0,079	y	n	n	192	35718023	35721177
23	0,036	0,493	0,057	y	n	y	272	35727882	35730942
24	0,029	0,278	0,059	y	n	n	149	35738048	35741484
25	0,025	0,765	0,050	y	n	y	421	35748570	35751302

Test	Accuracy [m]			lag ok? [y/n]			Time difference [ms]		
Nr	A	H	Q	A	H	Q	A	H	Q
26	0,050	0,751	0,045	y	n	y	48	35760526	35764077
27	0,055	1,287	0,060	y	n	y	35768230	35773317	35776164
28	0,042	0,731	0,059	y	n	n	212	35784921	35788005
29	0,060	0,774	0,068	y	n	n	35792576	35796135	35798969
30	0,034	0,487	0,096	y	n	n	67	35933185	35936093
31	0,023	0,269	0,065	y	n	n	130	35941950	35945153
32	0,044	0,498	0,096	y	n	n	35949008	35951162	35953738
33	0,043	0,709	0,064	y	n	n	1	35962266	35965276
34	0,099	0,608	0,046	y	n	n	35969446	35971997	31
35	0,058	0,959	0,064	y	n	n	35981254	35983855	35988897
36	0,032	0,741	0,042	y	n	n	213	35996040	55
37	0,045	0,134	0,046	y	n	y	206	36008606	192
38	0,032	0,537	0,033	y	n	n	79	36019316	103
39	0,062	1,210	0,057	y	n	y	36028086	36030764	170
40	0,028	0,424	0,063	y	n	n	303	36044694	36049425
41	0,031	0,678	0,051	y	n	n	183	36057195	36062028
42	0,059	0,808	0,060	y	n	n	36066549	36069416	36073320
43	0,073	0,611	0,069	y	n	n	36079026	36081737	36085812
44	0,062	0,805	0,054	n	n	n	36089969	36093758	36097604
45	0,038	0,739	0,070	y	n	n	32	36105603	36108777
46	0,047	0,621	0,060	n	n	n	36112800	36115929	36119422
47	0,100	0,703	0,064	n	n	n	36124407	36127524	36133029
48	0,049	0,751	0,064	y	n	n	36137125	36139844	36143846
49	0,062	0,693	0,067	y	n	n	36148679	36151431	36156379
50	0,017	0,054	0,099	y	y	n	203	36389464	36392577
51	0,054	0,253	0,069	y	n	n	36395534	36398512	36401313
52	0,021	0,349	0,056	y	n	n	349	36408063	36411209
53	0,025	0,044	0,051	y	y	n	195	445	36419433
54	0,017	0,047	0,060	y	y	n	306	282	36428874
55	0,021	0,043	0,058	y	y	n	420	347	36436894
56	0,021	0,625	0,055	y	n	y	273	36441407	36444430
57	0,025	0,048	0,050	y	y	n	199	126	11
58	0,030	0,060	0,054	y	y	n	84	121	36462779
59	0,046	0,479	0,078	y	n	n	563	36513670	36516316
60	0,025	0,759	0,044	y	n	y	185	36522460	290
61	0,028	0,040	0,057	y	y	n	188	230	36535939
62	0,055	0,200	0,068	y	n	n	36550360	36552887	36555710
63	0,024	0,067	0,086	y	n	n	263	36562959	36566465
64	0,034	0,583	0,057	y	n	n	516	36573764	36576619
65	0,027	0,052	0,049	y	n	n	338	36583299	36585967

Test	Accuracy [m]			lag ok? [y/n]			Time difference [ms]		
Nr	A	H	Q	A	H	Q	A	H	Q
66	0,033	0,073	0,054	y	n	n	408	36593076	36596823
67	0,033	0,068	0,063	y	n	n	253	36603893	36607137
68	0,034	0,055	0,055	y	n	n	134	36612519	36615803
69	0,078	0,060	0,063	n	n	n	36618470	36621689	36625531
70	0,067	0,050	0,054	n	n	n	36628836	212	36635537
71	0,051	0,052	0,063	n	y	n	36638556	169	36646614
72	0,052	0,056	0,062	y	n	n	10	36652350	36655197
73	0,032	0,349	0,095	y	n	n	86	36815172	36818596
74	0,059	0,564	0,090	n	n	n	36821991	36824678	36827402
75	0,038	0,522	0,083	y	n	n	43	36834619	36837531
76	0,035	0,484	0,097	y	n	n	72	36843916	36846685
77	0,049	0,359	0,084	y	n	n	36862249	36866574	36870347
78	0,085	0,662	0,102	n	n	n	36879427	36883711	36886935
79	0,052	0,456	0,086	y	n	n	36892353	36895438	36898256
80	0,046	0,398	0,101	y	n	n	36902204	36906058	36909986
81	0,060	0,433	0,110	y	n	n	36914713	36917420	36920668
82	0,049	0,443	0,096	y	n	n	36925517	36928384	36931358
83	0,075	0,966	0,086	n	n	n	36935712	36938784	36941725
84	0,066	1,088	0,095	n	n	n	36945640	36948421	36951260
85	0,101	1,113	0,096	n	n	n	36955552	36958153	36963150
86	0,078	0,992	0,100	n	n	n	36968155	36971182	36974176
87	0,095	1,092	0,090	n	n	n	36979177	36981925	36984940
88	0,078	1,244	0,104	n	n	n	37016047	37018579	37021499
89	0,092	1,078	0,099	n	n	n	37026262	37031067	37034257
90	0,069	0,183	0,090	y	n	n	37294506	37297562	37300535
91	0,062	0,082	0,090	n	n	n	37305753	37308624	37311504
92	0,050	0,283	0,083	n	n	n	37316808	37319806	37322800
93	0,027	0,242	0,066	n	n	n	66	37330156	37333400
94	0,050	0,224	0,100	n	n	n	37338610	37341493	37344479
95	0,043	0,673	0,080	y	n	n	10	37352028	37355333
96	0,042	0,741	0,066	y	n	n	11	37363488	37366240
97	0,045	1,583	0,080	n	n	n	37370824	37373768	37376730
98	0,059	0,629	0,064	n	n	n	37381227	37384237	37387305
99	0,057	0,611	0,063	n	n	n	37392085	37394821	37398118
100	0,047	0,680	0,055	y	n	n	37402546	37405880	37408698
101	0,035	0,826	0,048	y	n	n	94	37416165	37419474
102	0,016	0,698	0,062	n	n	n	260	37426031	37429005
103	0,035	1,057	0,066	y	n	y	169	37436090	37438949
104	0,080	0,394	0,056	y	n	n	37444569	37447592	37450504