Automation of High-accuracy Marking Tasks at MAX IV using the Quadrupedal Robot Spot

Sebastian Gulz-Haake

Nils Karlbrink Malmquist

LUND
UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6189
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2023

Abstract

When installing new research equipment at the synchrotron research laboratory MAX IV, a final positional accuracy of tens of microns is usually needed. To obtain that accuracy, initial reference points must be placed within 2 mm from the nominal target points in CAD, in a process referred to as bluelining. It is currently a manual process where points from a virtual environment are transferred to the real working environment and marked with the help of a laser tracker.

The purpose of this Master's Thesis is to explore the possibilities of automating the bluelining process using a 12 DOF quadrupedal robot equipped with a 6 DOF and gripper robotic arm together with a Leica AT403 external tracking system. The specific robot is the robot dog Spot from Boston Dynamics equipped with a Spot Arm. The system that has been developed consists of a platform carried by Spot's gripper containing the tracked reflector and a marking assembly, and of a software solution controlling the robot.

The proposed workflow starts with a localization step where Spot finds the tracker and orients itself in the tracker's frame. Once oriented, Spot positions itself within reach of a target point before positioning the marking device on the point using closed-loop feedback control. Once the point is reached, the operator actuates the marking device remotely after which Spot continues to the next point.

With this system, a marking accuracy within 2 mm from the target was achieved repeatedly in the real working environment at MAX IV. The process time was found to be around 1 minute and 30 seconds per point, which is comparable to a human operator.

From the results and data gathered during the project, it can be concluded that the proposed solution shows promise regarding both accuracy and autonomy. Furthermore, it can be concluded that similar solutions have the potential to be expanded into other areas and automate similar processes, such as marking processes for equipment installation at other types of research laboratories, as well as for booth setups at exhibitions and fares, and at construction sites when marking out mounting locations.

Acknowledgments

We would like to thank the Department of Automatic Control at LTH and the Center of Construction Robotics for giving us the opportunity to work with this project, providing us with continuous access to the robot Spot, and Cognibotics for providing us with the rest of the needed equipment as well as development space.

We would like to express our deepest gratitude to our academic and industrial supervisors, Björn Olofsson and Mathias Haage, respectively, for their valuable input and continuous support throughout the project. We also want to thank our assistant supervisor Alina Andersson, Research Engineer at MAX IV, for her dedication to the project, investing both time and resources vital for the end result.

Furthermore, we want to thank Alexander Pisarevskiy, Research Engineer at Automatic Control, for helping us with electrical components, and Johan Lauri at Cognibotics for helping us with office space. At Cognibotics, we also want to thank Klas Nilsson and Ola Nilsson for sharing their knowledge of robotics and Spot in particular, and Fredrik Malmgren for enduring our company.

Lastly, we would like express our gratitude to our examiner Anders Robertsson, for giving us the opportunity to work with this interesting thesis project.

The thesis has been carried out within the research projects "Semantic Mapping and Visual Navigation for Smart Robots" (RIT15-0038) with financing from the Swedish Foundation for Strategic Research (SSF), and "Buster - the Builder's Best Friend", 2022-2023, funded by Svenska Byggbranchens Utvecklingsfond (SBUF) 14115.

It has also been made possible by the now finished projects "Automated production in concrete construction", 2018-2019, funded by SBUF 13610 and "Utmätnings- & utsättningsrobotar - Ökad precision och informationsåterkoppling", 2020-2022, funded by SBUF 13967.

Contents

1

Introduction

1.1 Background

As the possibilities to program mobile robots to navigate and act autonomously increase, new use cases in the industry open up. They can for example be used to replace humans in labor intensive, dangerous, and repetitive tasks.

The work reported in this thesis was made possible by the Center for Construction Robotics at Lund University [Lund University, Faculty of Engineering, 2023], Cognibotics AB [Cognibotics AB, 2023] and MAX IV [Lund University, MAX IV, 2023a]. The Center for Construction Robotics contributed with access to the Spot quadruped robot [Boston Dynamics, 2022b], Cognibotics contributed with supervision and access to tracking equipment as well as a workplace, and MAX IV contributed with a usage scenario, supervision and access to the MAX IV facility with tracking equipment onsite. We believe the techniques developed within this thesis generalize well to accurate quadruped making tasks in other environments, such as at construction site, where the work includes labor- and time-intensive tasks such as laser scanning and marking out positions for holes and interior walls.

The use case that will be the focus of this thesis is installing new scientific equipment at MAX IV, a synchrotron laboratory in Lund, Sweden. This is done with a process called bluelining where a CAD model is transferred to the floor by measuring and drawing with markers, aided by a laser tracker to achieve the needed accuracy which in this case is around 2 mm. By utilizing Spot's capability of autonomously navigating, adapting to, and interacting with changing environments, together with some proprietary hardware, this task could be automated while obtaining the same accuracy.

MAX IV

MAX IV Laboratory is a Swedish national laboratory located in Lund. It houses two synchrotron rings at 3 and 1.5 GeV, respectively, as well as one linear accelerator [Lund University, MAX IV, 2023c]. From the synchrotron rings, several beamlines extend dedicated for different types of scientific experiments. In Figure 1.1

a schematic over MAX IV with its two synchrotron rings, linear accelerator and currently operating beamlines can be seen.

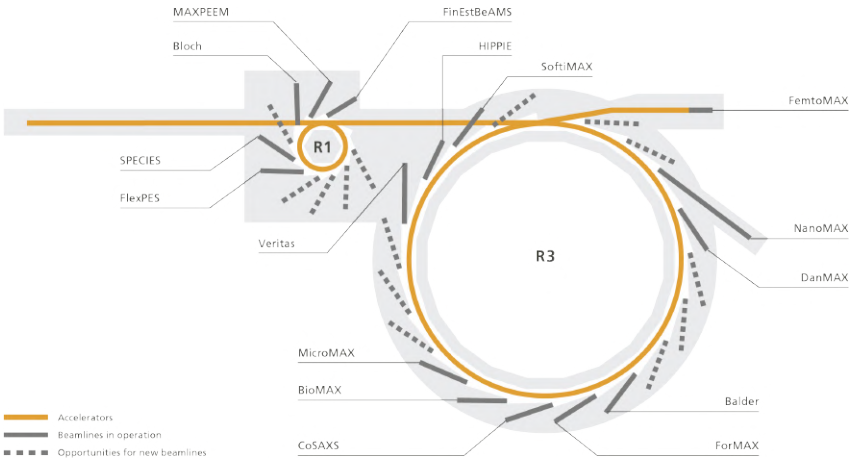


Figure 1.1 Schematic over MAX IV Laboratory, showing the two synchrotron rings, denoted R1 and R3, as well as the linear accelerator shown horizontally in the figure, and the currently operating beamlines [Frida Nilsson/MAX IV, 2023].

A beamline [Lund University, MAX IV, 2023b] is the instrumentation that carries beams of synchrotron radiation to an end station. It consist of various parts, equipment and instruments placed in a perfect straight line. When building a new beamline, precision and accuracy are therefore important.

Bluelining

Bluelining is the process of transferring points from CAD to the real workspace at MAX IV. It is one of the first stages involved in the process of installing research equipment at the right place. Most of the equipment is placed along the beamlines and because of the narrow light beams, it has to be placed coaxial with an accuracy that allows it to be adjusted to within tens of microns from the nominal position.

Bluelining is facilitated by a physical network consisting of nodes, which are fixed positions for reflectors; see Figure 1.2a, that are placed throughout the facilities. Reflectors are placed in the nodes and their positions are measured with laser trackers, which allows a virtual representation of the network to be constructed from the measurement data. This makes it possible to design and place test equipment in CAD software, obtain marking data (e.g., mounting points for equipment) in the virtual network and then transfer those points to the real world. To achieve this, a laser tracker is placed in the network and located by measuring its distance and angles to a minimum of three network nodes with reflectors placed in them, creating a

virtual twin of the tracker. The node's location in the real network then corresponds to the node's location in the virtual network.

(a) A Leica reflector placed in a MAX IV network node.

(b) A Leica reflector placed in the bluelining tool.

Figure 1.2 Leica reflectors during different steps in the bluelining process.

When the tracker is located in the network, the actual bluelining task can begin. It is done by placing a reflector in a handheld machined holder, see Figure 1.2b, hereafter called the bluelining tool, and moving it to the correct location in the network with the help of measurement data streamed from the tracker. The data can be given to the operator either as absolute coordinates in (x,y,z), or relative coordinates showing distance to the goal point. When a satisfactory location is reached, the reflector is removed from the bluelining tool and a permanent marker is used to make a mark through a hole located at the tool's vertical symmetry axis, which is coaxial with the reflector's center axis. The accuracy obtained by this method is around 2 mm, which is sufficient.

Previous Work

Several studies regarding positional tracking and autonomous marking have been conducted in the past. For example, in [Inoue and Ohmoto, 2012] an omnidirectional mobile robot was constructed and utilized to create markings in a construction site environment within 2 mm of the references, while relying on a Laser Range Finder and a Total Station for positioning.

Furthermore, in [Tsuruta et al., 2019] an automated mobile robotic system for marking free access floors was developed. The developed robot managed to achieve a mean accuracy of 2.3 mm over 182 continuous points using a Laser Positioning Unit for tracking.

There have previously been two Master Theses carried out regarding the task of using mobile robots for automatic bluelining at MAX IV. In [Klinghav, 2021], the author used a TurtleBot with a 3D printer attached to carry out the task. The robot was controlled by a previously developed ROS integration, moving it coarsely to the wanted position, and the fine positioning was carried out by the 3D printer by translating the coordinates to G-code and sending the G-code to the printer. The reflector was mounted on a near vertical axis with the permanent marker. The setup was successful in completing the task with high precision, but lacked the needed accuracy. This was thought to be mainly because of tilt of the marker-reflector axis. The speed was also low at an estimated 30 points in 4 hours, mainly because of the control method used where the robot moved in discrete steps.

In [Patil, 2022], a custom-built Cartesian robot was used. It was controlled over Modbus and functioned much like a 3D-printer. This robot in itself was not mobile, but was dependent on an external moving platform currently under development, as presented in [Andersson and Shahin, 2022a], and was otherwise constrained to a 300×300 mm^2 reaching space. It achieved both high precision and high accuracy, with a positioning resolution of 60 microns, but was lacking in speed for the fine positioning and was restricted in reach without the moving platform.

1.2 Research Questions

The previous work indicates that autonomous bluelining as a concept shows promise, but that there are limitations when using small wheel-based robots and discrete iterative positioning for the task. While the accuracy obtained can be high, the process times are long. The robots are also constrained to flat, even floors without obstacles. Even if some of these limitations can be worked around at a facility like MAX IV, where the floor is flat and even, they make it complicated to extend the application into other areas, such as construction sites or other industrial facilities where the same level of flatness and clean working environment cannot be ensured. Being strictly ground bound also presents potential problems to both the bluelining process and similar marking processes in other application areas, which can also include placing marks on walls and other surfaces. From this background, the following research questions were investigated in this thesis:

- What are the limitations for using Spot equipped with the Spot Arm for high precision and accuracy operations?
- How precise can Spot be maneuvered relative to previous solutions for bluelining, and how well can Spot be adapted to a constrained movement scheme, while still retaining the benefits of its autonomous features?
- Can the solution developed in this thesis be extended into other application areas, and in that case, what consideration has to be taken into account in each case?

1.3 Technical Objectives

The technical goal of this thesis project is to have Spot autonomously navigating around the workspace at MAX IV to assist with the floor-marking process for equipment installation. This should be done with the help of a Leica laser tracker and a Leica reflector, and should be achieved without losing the line of sight between the laser tracker and the reflector. When Spot has reached a position close to the desired marking point, the Spot Arm with attached platform should be used to find the exact position and mark the point corresponding to the same point in CAD. To achieve this, the following intermediate steps should be accomplished.

- Design and development of a platform containing a Leica reflector and the bluelining marking device that can be picked up and carried by the gripper at the end of the Spot Arm.
- Trajectory and orientation planning to get to the desired point within a reasonable time frame without blocking or losing the line of sight.
- Positioning of the carried platform with an accuracy of 2 mm and actuation of the bluelining marking device.
- Communication with Spot to control its movements to make it follow planned trajectories, while retaining its autonomous navigation features such as obstacle avoidance, and actuate the bluelining marking device by using the available API.
- Redundancy in the form of a solution in the case of losing line of sight to the laser tracker.

1.4 Outline

- ***Chapter 2*** presents the theoretical background needed to understand the system and methodology.
- ***Chapter 3*** presents the developed system, including Spot and the tracker.
- ***Chapter 4*** presents the developed bluelining workflow with Spot step by step.
- ***Chapter 5*** presents the results from the main task, as well as the subsystems' performance and limitations.
- ***Chapter 6*** discusses the results and implementation.
- ***Chapter 7*** concludes the report with a brief conclusion and possible future extensions.

2

Theory

2.1 Positioning and Frames

This section covers the basic concepts and terminology regarding transformations used throughout the report, and is based on [Lynch and Park, 2017], if not otherwise stated.

Homogeneous Transformation

Homogeneous transformations are used to relate different frames of reference to each other and to transform points from one frame to another, see Figure 2.1.

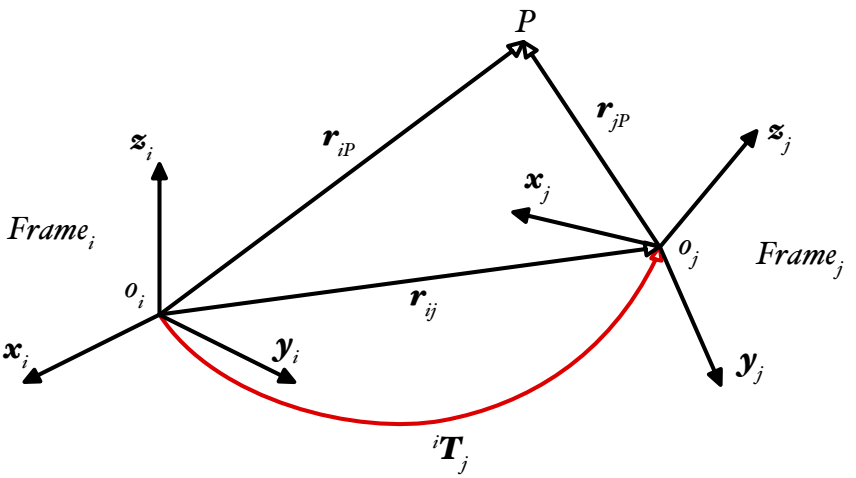


Figure 2.1 Homogeneous transformation from one frame to another.

They are linear transformations and can be expressed using matrices. A homogeneous transformation matrix is a 4×4 matrix that represents rotation and translation as

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \tag{2.1}$$

where $\mathbf{R}$ is a 3×3 rotation matrix, such as e.g., rotation θ around the z-axis

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

and $\mathbf{t}$ is a 3×1 translation vector. $\mathbf{H}$ is invertible such that

$$\mathbf{H}^{-1} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{R}^\intercal & -\mathbf{R}^\intercal\mathbf{t} \\ 0 & 1 \end{bmatrix} \tag{2.3}$$

Obtaining the Transformation Matrix between two Frames

To obtain the homogeneous transformation matrix from one frame to another for a point, $\mathbf{R}$ and $\mathbf{t}$ need to be determined. For a physical system this can be done by sampling a set of n points, where $n > 2$, in both frames simultaneously, see Figure 2.2, resulting in one set per frame, $\mathbf{P}$ and $\mathbf{Q}$.

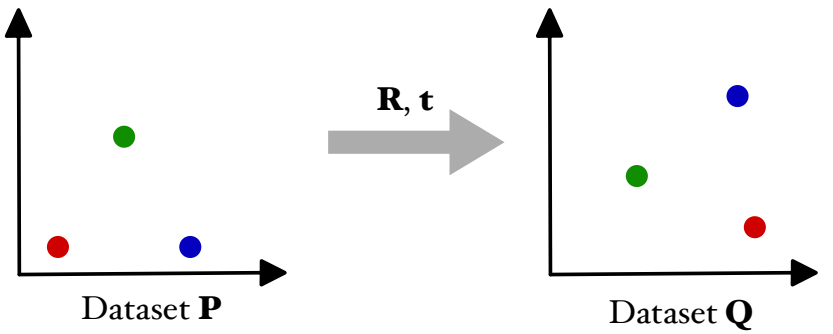


Figure 2.2 The same set of points sampled in two different frames.

$\mathbf{P}$ and $\mathbf{Q}$ can then be used to determine $\mathbf{R}$ and $\mathbf{t}$ by applying Kabsch algorithm [Kabsch, 1976]. First the centroids of the two sets $\mathbf{P}$ and $\mathbf{Q}$ are calculated as

$$\begin{aligned} \mathbf{P}_m &= \frac{1}{N}\sum_{n=1}^{N}\mathbf{P}_n \\ \mathbf{Q}_m &= \frac{1}{N}\sum_{n=1}^{N}\mathbf{Q}_n \end{aligned} \tag{2.4}$$

Then the centroids are moved to the origin in the respective frame, resulting in $\mathbf{P}'$ and $\mathbf{Q}'$ as follows

$$\begin{aligned} \mathbf{P}' &= \mathbf{P} - \mathbf{P}_m \\ \mathbf{Q}' &= \mathbf{Q} - \mathbf{Q}_m \end{aligned} \tag{2.5}$$

The covariance matrix $\mathbf{C}$ can now be calculated as

$$\mathbf{C} = \mathbf{P}'\mathbf{Q}'^{\mathsf{T}} \tag{2.6}$$

and the singular value decomposition of $\mathbf{C}$ is computed for

$$\mathbf{C} = \mathbf{U}\Sigma\mathbf{V}^{\mathsf{T}} \tag{2.7}$$

$\mathbf{R}$ can now be calculated as

$$\mathbf{R} = \mathbf{V}\mathbf{U}^{\mathsf{T}} \tag{2.8}$$

and the translation vector $\mathbf{t}$ is given by

$$\mathbf{t} = \mathbf{Q}_m - \mathbf{R}\mathbf{P}_m \tag{2.9}$$

2.2 Automatic Control

Automatic Control can be described as the technology used to control systems and processes in order for them to behave in a desired way and to produce the desired results. By using mathematical descriptions and approximations of physical processes, models and algorithms can be constructed to control these processes, which can be everything from auto piloting an aircraft to adjusting hundreds of process parameters in an industrial plant [Hägglund, 2019]. This section covers closed-loop systems and PID control, which are both used in this thesis work.

Closed-loop Systems

Compared to open-loop systems, where the output of the system depends entirely on independent input references and has no effect on the control action, closed-loop systems adjust the input based on the output to steer it towards a desired condition [Hägglund, 2019]. This means that the control action in a closed-loop system is based on the output and closed-loop systems are therefore also known as feedback systems. The schematic of a simple closed-loop system can be seen in Figure 2.3, where r denotes the reference value, also known as the set point, u denotes the control signal that is fed into the process and y denotes the output that is measured and fed back into the system to adjust the control action accordingly.

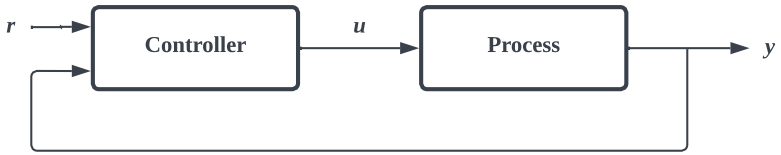


Figure 2.3 Schematic of a simple closed-loop system with reference value r, control signal u and output y.

PID Controller

A proportional-integral-derivative (PID) controller is a control system that uses feedback to adjust the behavior of a system in order to achieve a desired outcome. According to [Hägglund, 2019] it is the most commonly used controller and is used in, e.g., industrial and technological applications to control processes and maintain stability. A PID controller consists of three components: a proportional (P), an integral (I), and a derivative (D) term. The proportional term is based on the current error between the desired output, also known as the set point, and the actual output of the system. The integral term is based on the accumulation of past errors over time, and the derivative term is based on the rate of change of the error. These three terms are combined to calculate the output signal of the PID controller, which is used to adjust the system's behavior and reduce the error, see Figure 2.4. In a general, simplified scenario, the proportional term provides a quick response to the error, the integral term helps to eliminate any steady-state error, and the derivative term provides damping to prevent overshoot and oscillation.

[Hägglund, 2019] tells us that the strength of each of these terms can be adjusted through the use of parameters, known as gains, which determine the weight of each term in the final output. By carefully adjusting the gains, the PID controller can be tuned to achieve the desired performance. It is possible to adjust the gains to the extent of setting one, or more, of the gains to equal zero, to completely disregard that part of the controller when calculating the output signal. By, e.g., adjusting the gain of the D part to zero, the controller will end up as a PI controller.

The P Part. The P part can be written as

$$K_p e(t) \tag{2.10}$$

where K_p is the proportional gain and $e(t) = r(t) - y(t)$ is the computed error at time t. A high K_p will provide a quick response to the error, but increases the risk of the output overshooting the set point. A low K_p will decrease the risk of overshoot, but provides a slower response. There will always be a stationary error with only a P controller, which can be reduced, but not removed by a higher K_p.

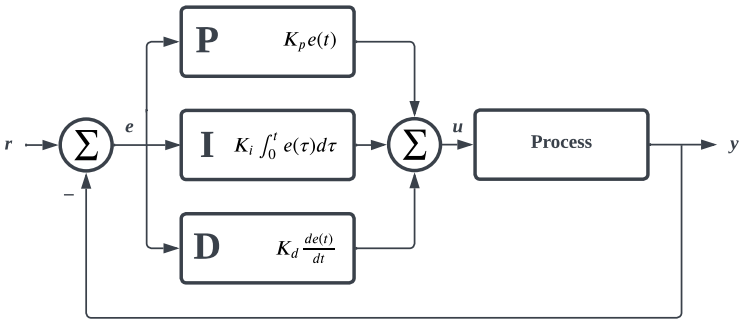


Figure 2.4 PID controller schematic showing how the different parts affect the control signal.

The I Part. The I part can be written as

$$K_i \int_0^t e(\tau)d\tau \tag{2.11}$$

where K_i is the integral gain and the integral term is the accumulated error over time. A high K_i will quickly eliminate any steady-state error, but also increases the risk of overshoot and integral windup.

The D Part. The D part can be written as

$$K_d \frac{de(t)}{dt} \tag{2.12}$$

where K_d is the derivative gain and the derivative term is the rate of change of the error. A high K_d will quickly dampen the effect of sudden changes since sudden changes typically lead to large derivative of error and thus, a large contribution from the D part. However, the D part also adds complexity to the system. Since the D part extrapolates the current slope of the error, it is, e.g., highly affected by noise in the process variables. Noise, in this case, means small, random, and rapid changes in the system which in turn causes rapid changes to the error.

For the practical implementation of PID controllers, readers are referred to e.g., [Åström and Hägglund, 1995].

2.3 Concurrent Programming

This section covers concurrent programming, which is used in the software development part of this thesis work, and the Python GIL, which limits how Python handles concurrent tasks.

Concurrent Programming

Concurrent programming is a type of programming that, according to [Goetz et al., 2006], allows multiple tasks to be performed simultaneously. This can be achieved in a number of different ways, depending on the programming language and run time environment. For example, some languages and run times provide support for threads, which are independent units of execution that can run concurrently within a single program. Other languages and run times provide support for parallelism, which allows multiple processes or threads to be executed simultaneously on different CPU cores or processors.

The goal of concurrent programming is to enable programs to take advantage of modern hardware architectures that have multiple cores or processors, and to improve the performance and responsiveness of programs by allowing them to perform multiple tasks simultaneously. This can be especially useful for applications that need to perform multiple independent tasks concurrently, such as handling multiple client connections in a server, or executing multiple independent computations in a scientific application.

The Python GIL

In Python, the Global Interpreter Lock (GIL) [Python Software Foundation, 2023] is a mechanism that prevents multiple native threads from executing Python bytecode at once. This means that only one thread can execute Python code at a given time, even if there are multiple cores or CPUs available.

One way to work around the GIL is to use the threading module [Python Software Foundation, 2022b]. This module allows for the creation of multiple threads, and each thread can run Python code concurrently. However, because of the GIL, the threads will not actually run in parallel, and they will only provide a performance benefit if they are performing I/O-bound or CPU-bound operations that release the GIL periodically.

Another option is to use the multiprocessing module [Python Software Foundation, 2022a], which allows for the creation of multiple processes instead of threads. Because each process has its own GIL, this allows code to take advantage of multiple cores or CPUs, and it can provide a significant performance boost for CPU-bound and I/O-bound operations. However, it is important to note that the processes do not share memory, so some kind of inter-process communication (IPC) mechanisms has to be used to coordinate their work.

3

System

This chapter describes the solution on a system level. Sections 3.1 and 3.2 cover the preexisting equipment and systems used, while Sections 3.3 and 3.4 present the hardware and software that have been developed as part of the thesis. Section 3.5 shows how all parts are integrated to form the full system. 3D files for the hardware and the source code for the software developed for this thesis can be accessed on the public repository *https://github.com/sebastiangulzhaake/spot-bluelining* [Gulz-Haake and Karlbrink Malmquist, 2023b].

3.1 Spot

Spot is a quadrupedal robot, see Figure 3.1, from Boston Dynamics, released in 2019. The base robot can be programmed to navigate autonomously and is mainly used to perform inspections and data gathering missions, often in environments that are either dangerous and/or hard to navigate for humans.

Figure 3.1 Spot from Boston Dynamics in the configuration used in this thesis.

Hardware

The specifications provided by Boston Dynamics [Boston Dynamics, 2022b] state that the base robot has four legs, each with 3 degrees of freedom, resulting in 12 degrees of freedom for the entire base robot. On each side of the body, the robot is equipped with camera arrays for image capturing, obstacle avoidance, depth sensing and autonomous navigation.

Apart from the base robot, there is a range of extensions called *payloads* that can be attached to the base robot to extend its capabilities. The robot used throughout this thesis project is fitted with two payloads, the Enhanced Autonomy Payload (EAP) [Boston Dynamics, 2022i] seen in Figure 3.2b, and the Spot Arm [Boston Dynamics, 2022h] seen in Figure 3.2a. The EAP consists of the Spot Core computer which allows the user to deploy programs locally on the robot and a LiDAR which gives Spot increased range and resolution for autonomous data collection, mapping, and navigation. The Spot Arm is a 6 degrees of freedom robotic arm with an end effector in the form of a gripper. The arm, which is used extensively in this system design, provides Spot with the capability to interact with objects in the world and carry out tasks relying on physical manipulation.

(a) The Spot Arm that extends Spot's manipulation and interaction capabilities.

(b) The Spot EAP with LiDAR and the Spot Core computer.

Figure 3.2 The two payloads the particular Spot used in this thesis was equipped with.

Software

Spot runs on proprietary software that is not directly accessible for the user, meaning control access granted to the user is restricted. Therefore, low-level control of individual motors as well as fine control of Spot's behavior are not possible, resulting in the system largely having to be treated as a black box throughout this thesis.

The limited control of low-level functions complicates the development of high-accuracy arm control. To achieve accurate positional control of the Spot Arm for example, it would be beneficial if Spot's body could be fixated during arm move-

ment. As it is now, Spot continuously balances when moving the arm to counter the changes in center of gravity. This causes Spot's local frame to move relative to the global frame, changing the frame of reference for global positioning during operation.

To aid in developing applications for and controlling the robot, Boston Dynamics provides an SDK containing an API protocol definition and the Spot SDK repository where all of the Spot SDK code, for example, a Python client library is hosted [Boston Dynamics, 2022e].

The Python library utilizes the Spot API and lets users write applications for controlling Spot by sending commands, requesting information from sensors and system estimations, and creating and integrating external payloads. The Spot API follows a client-server model, where client applications communicate with services running on Spot over a network connection.

Furthermore, Boston Dynamics specify that gRPC is the application-level protocol used by the Spot API [Boston Dynamics, 2022c]. gRPC is a cross-platform open source Remote Procedure Call framework that supports a broad set of programming languages and environments and it uses Protocol Buffers as input and output message format. Protocol Buffers are just as gRPC implemented for a broad set of programming languages and environments, and typically used when developing programs communicating with each other over a network [Google, 2023].

Client applications can be run on a variety of systems, as long as they can establish a network connection to Spot, such as a tablet or a laptop computer [Boston Dynamics, 2022g]. Throughout this project, a laptop computer is used as the client application to execute the bluelining program.

Geometry and Frames

The documentation about Spot's geometry and frames [Boston Dynamics, 2022f] states that Spot uses a collection of frames to orient itself and to be able to carry out tasks. At the root, there is the *body* frame, which is aligned with Spots body, with the origin always fixed at the geometric center of the robot. Related to this frame are other frames, for example the *hand* frame, which is mechanically linked to the *body* frame through the joints in the arm and estimated with measured joint angels at any given moment.

The specifications also provide two inertial, or global frames, the *odometry* frame and the *vision* frame. Both of them have an origin and initial rotation matching that of the *body* frame when booting the robot. In contrast to the *body* frame that is fixed to the robot, the *vision* and *odometry* frames move relative to the robot's movements to provide an estimate of the fixed location in the world, relative to where the robot is booted. The *odometry* frame is estimated using the kinematics of the robot and measurements such as encoder and actuator data, while the *vision* frame is calculated using visual analysis of the world through the robot's cameras and the robot's odometry. Both frames are used to translate global coordinates, ro-

tations and velocities to local ones, and vice versa. They provide Spot with a way to localize and navigate itself autonomously, but since the global frames are continuously estimated, the accuracy depends on the estimate which can vary over both space and time.

Networking

There are several ways of communicating with Spot. The default one, as stated by Boston Dynamics, is to connect to Spot as an access point, allowing data to be sent bidirectionally with gRPC between the connected device and Spot [Boston Dynamics, 2022c]. This networking configuration is, for example, used when Spot is operated manually with the tablet. There is also the option of connecting Spot to a WLAN. Lastly, Spot has an Ethernet port in the rear which can be utilized for a wired connection.

In addition, it is possible to communicate with payloads, for example with services running on the Spot Core, mentioned earlier, or other external systems, using port forwarding to route the communication through Spot's network as described in [Boston Dynamics, 2022d].

3.2 Positional Tracking

Positional tracking is a technology that can be used for both tracking and measuring during construction, calibration, manufacturing, as well as making adjustments and for quality control. In this thesis, the focus is mainly on the tracking part, but static measuring is also used for orienting Spot and for validating the system performance.

Throughout this thesis project, two positional tracking systems have been used. These are the Leica AT403 laser-based tracking system [Leica Geosystems, 2017], used at MAX IV and seen in Figure 3.3a and the Krypton K610 camera-based tracking system [Krypton Electronic Engineering, 2003], used during development and testing, and seen in Figure 3.3b.

Laser-Based Tracking Systems

Laser-based tracking systems usually consist of a tracking station and a reflector. The tracking station, whose position is initially measured and thereafter considered known, emits a laser beam that is reflected by the reflector, providing an accurate 3 degrees of freedom positioning. The technology can be combined with a camera and diodes, effectively combining laser and camera tracking, to provide 6 degrees of freedom positioning.

Camera-Based Tracking Systems

Camera-based tracking systems usually consist of a base unit with several cameras that tracks different objects, e.g., LEDs or body movements. In the case of using LEDs, the cameras in the base unit triangulate the LED's position to provide 3

degrees of freedom positional data in the case of one LED and 6 degrees of freedom with three or more LEDs.

(a) The Leica AT403 laser based tracker.

(b) The Krypton K610 camera based tracker.

Figure 3.3 The two trackers used in this thesis work with fiducials mounted on them.

Limitations

Depending on which type of system being used, different limitations are imposed on the tracking. Some examples of those limitations are:

- ***Update Rate***
  The update rate of the tracker decides the tracking resolution in the time domain and limits how fast moving targets can be tracked.

- ***Streaming Rate***
  The streaming rate limits how fast the measurements can be streamed to the operator and control program. If the streaming frequency is lower than the update rate, the full resolution of the tracker cannot be utilized for real-time control.

- ***Angle Offsets***
  Different tracking systems have different opening angles between the tracker and the reflector or LED(s) that they can accommodate without losing tracking. For laser-based tracking systems, there are several different types of reflectors that can accommodate different angle offsets between the line of sight axis of the tracker and the reflector. For camera-based systems, the angle of the LED(s) relative to the camera sensors' axes is also limited.

- ***Distance***
  All trackers have a minimum and maximum distance where they provide accurate measurements. This is often a trade off. For laser-based systems, longer distances can cause less spatial resolution, but accommodate higher speeds, since the tracker needs to move less for the same target movement. Laser-based systems usually have a longer tracking distance compared to camera-based systems.

- ***Tracking Envelope***
  Most camera-based systems are fixed and work with a tracking envelope comprised of the cameras' field of view. Laser-based systems on the other hand can often rotate and cover 360 degrees.

Leica

The Leica AT403 used at MAX IV for the implementation of this thesis project is one of two trackers that is used during the manual bluelining process, the other being a Leica AT401. The original plan was to be able to use either the AT403 or the AT401 interchangeably. However, this turned out to be a complicated approach beacuse of differences between the systems. During testing it became clear that the AT401 lacked the capability of providing accurate continuous measurements, rendering the control algorithm unable to efficiently reach the goal point.

The user manual for the AT403 [Leica Geosystems, 2017] states that the measurement range is 360° horizontally, ±145° vertically and between a distance of 0.8 and 160 m. The distance accuracy is ±10 µm for 0.8 m to 80 m and the angular accuracy is ±15 µm + 6 µm/m for the same range. Using a RRR or BRR, which are the two reflector types used in this thesis, the maximum entry angle is ±30°. The AT403 supports continuous measurements and can stream measured data at a rate of 10 Hz. However, the actual sampling of data is according to [Lippitsch, 2018] 5 Hz due to the quasi-dynamic (ADM) distance measurement technology used.

The tracking data are streamed via UDP from the computer connected to the tracker to the receiving operator computer running the bluelining program and controlling Spot.

Krypton

For continuous testing and implementation purposes, a Krypton K610 [Krypton Electronic Engineering, 2003] was used for tracking during development of the

bluelining workflow. It is a camera-based tracking system with three cameras tracking one or several pulsating IR LEDs. It is a stationary system with a fixed envelope of 17 m^3 distributed over three zones as shown in Figure 3.4. Zone I has a single point accuracy of ±60 µm + 7 µm/m, zone II has an accuracy of ±60 µm + 17 µm/m and zone III has an accuracy of ±130 µm + 17 µm/m. The maximum angle between the LED(s) and the cameras' axes is stated to be ±30° for reliable measurements. The K610 supports continuous measurements at 1000 Hz, and can stream measured data at 100 Hz with the provided software.

The tracking data are streamed via TCP from the computer connected to the tracker to the receiving operator computer running the bluelining program and controlling Spot.

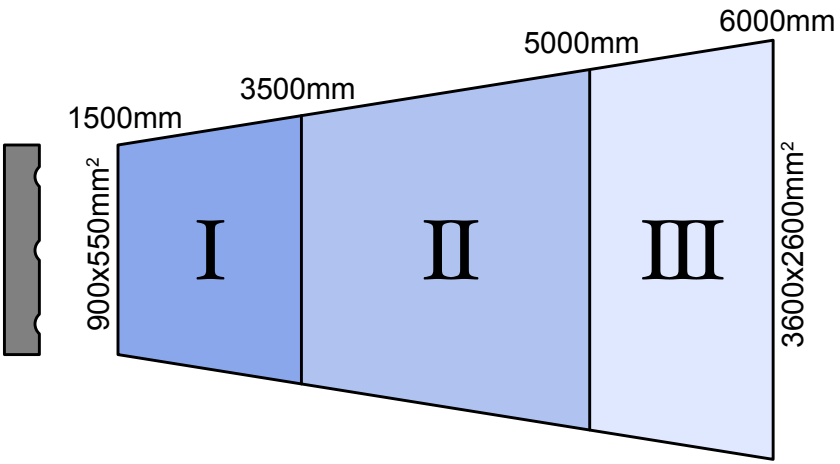


Figure 3.4 Top view of the tracking envelope of the Krypton K610, showing the three accuracy zones.

Node Network at MAX IV

When the tracker is located in the physical network defined by the nodes at MAX IV, any arbitrary point in this network can be expressed in the tracker's frame, which can be set up according to the user's need. For the purpose of this thesis, the frame is gravity aligned, with the origin located in the center of the tracker. The x- and z-axes extend parallel to the floor with the y-axis pointing upwards in a right-handed Cartesian coordinate system.

Special Considerations

When working with physical applications, it is important to remember that in the real world, nothing is truly orthogonal or perfectly aligned. In the scope of this

thesis, special consideration has to be taken when dealing with the movements of the robot and when orienting it in the tracker's frame that is based on the physical network at MAX IV. This has led to special measurements being taken, such as relying on a simple control method that compensates for errors in orientation, using velocity vectors rather than exact coordinates for control and implementing a measurement sequence that compensates for alignment differences between the floor and the network.

Furthermore, when developing the system, it was found that the depth camera in Spot's gripper as well as the depth cameras on the front of the robot interfered with the camera-based tracking systems, since it contains a pulsating IR LED with similar frequency to the measurement LEDs. Surprisingly, the gripper camera also interfered with the laser tracking, possibly because of the reflective surface containing the depth-camera array. To handle this, all interfering depth cameras were covered during operation.

3.3 Marking Device

The idea for the marking device was to design an assembly containing an electrically actuating pen tip that can be picked up and carried by Spot's gripper. This marking device has to be stiff enough to provide repeatable results, while being lightweight enough to be carried and moved around by Spot in an agile way.

Mechanical Design

The developed marking device, see Figure 3.5, consists of a base plate with an attached handle that allows Spot to pick it up, hold it, and place and drag it along the floor. The handle is molded from a 3D model of Spot's gripper and is secured both longitudinally and laterally with the help of the gripper's geometry, while still allowing some flexibility around the lateral axis. The dimensions of the base plate are 90×200 mm^2, providing a stable platform for the assembly and extends below Spot's gripper. This is done to increase the range of motion while keeping Spot's body in a stable position.

The actual marking assembly consists of a suspended collet, attached with two arms to two 5V solenoids with return springs, see Figure 3.6 for a section view. The collet houses the tip part of an Artline 70 permanent marker, which screws off the rest of the pen casing. Above the collet, the reflector is mounted on the same axis in a holder with a magnet at the bottom and secured in place with a halo-like support. The physical prototype can be seen in Figure 3.7a.

To further simplify the bluelining procedure, another design that could house a whole Artline 70 permanent marker was derived. This design increases the up time of the procedure since the pen can be used for longer periods before it needs to be changed, compared to the tip that dries out after around 15 minutes or 10 marks. A direct cause of this design is that the reflector is placed 105 mm higher from the

ground, increasing potential tilt and inclination problems. This design can be seen in Figure 3.7b.

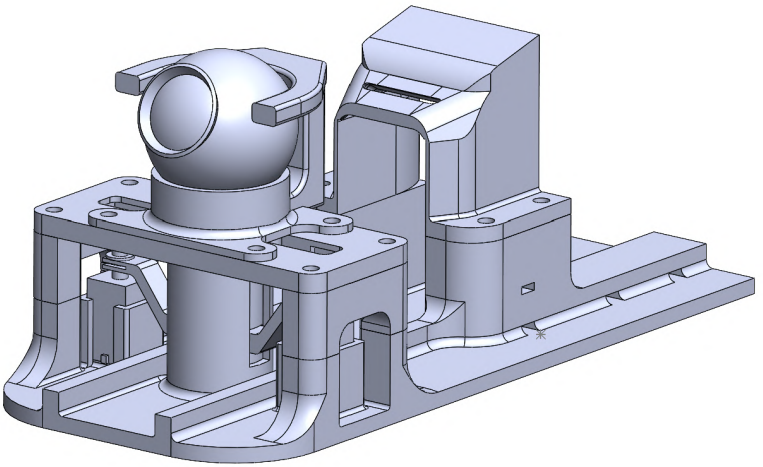


Figure 3.5 CAD assembly of the marking device, showing the handle, the mounting and locking of the reflector, and the general design of the device.

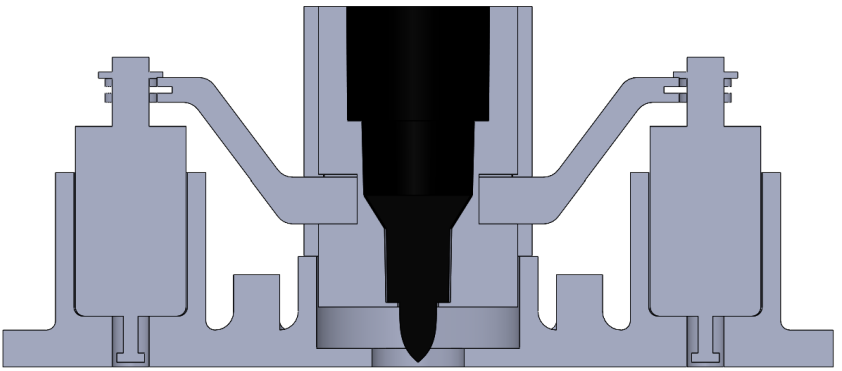
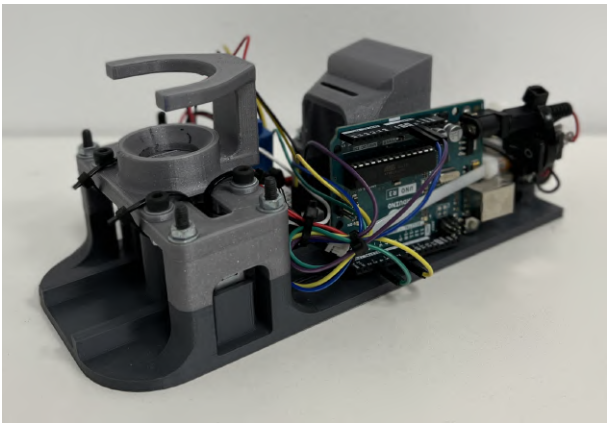
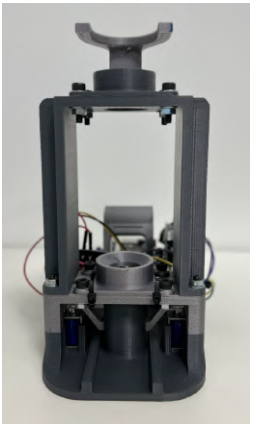


Figure 3.6 Section view of the marking assembly, showing how the collet housing the pen tip is connected to the solenoids with two arms.



(a) The first prototype of the marking device with a low reflector position. This configuration only accommodates a pen tip due to spacial constraints, resulting in a short operating period of around 15 minutes or 10 marks before replacing the tip when it has dried out.



(b) Marking device with a high reflector position. This configuration accommodates a whole pen, increasing the up time.

Figure 3.7 The two prototypes.

Electrical Design and Actuation

The marking operation is performed by actuating the solenoids, which pulls the collet down until the pen tip reaches the floor, leaving a permanent mark on the floor.

In the first design, the solenoids were actuated by pressing a button mounted on the front of the assembly. The drawback of this design was that the operator had to be close to Spot to actuate the solenoids, potentially blocking the line of sight to the tracker, and risking moving the marking platform when pressing the button.

A second design was derived where the solenoids are actuated with an IR remote. The IR signal is received by an IR receiver connected to an Arduino that controls a relay that the solenoids are connected to.

3.4 Bluelining Software Architecture

Bluelining Program

The developed software solution for the bluelining program is written in Python. The program implements four threads running concurrently, as covered in Section 2.3, for controlling Spot while also receiving tracker data, querying robot state and waiting for keyboard input. Shared data are stored in three different monitors protected by locks to avoid, e.g., race conditions such as check-then-act situations. The architecture of the program can be seen in Figure 3.8.

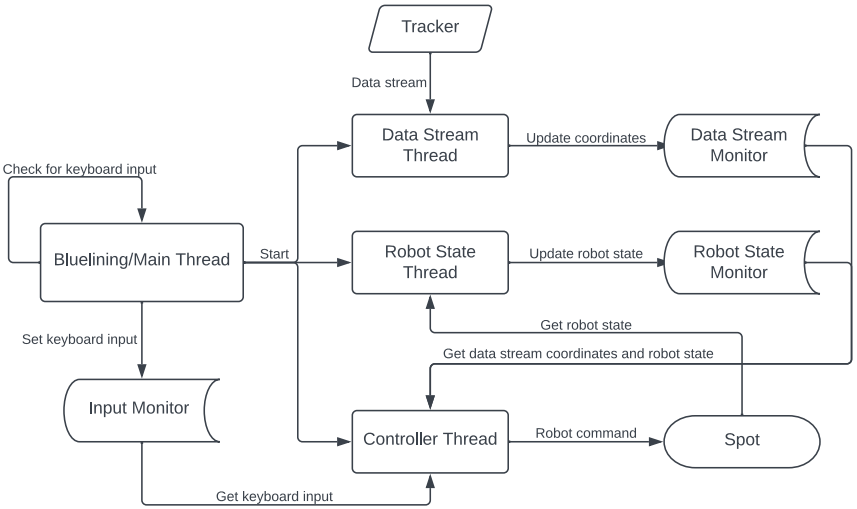


Figure 3.8 The bluelining program architecture, showing the different threads and monitors, and how these interact with Spot, operator input, and the data stream from the tracker.

Main Thread. The main thread creates all monitor objects and starts all other threads. After starting the threads, the main thread continues to wait for keyboard inputs. When an input is received, the value is saved to the input monitor to be accessible for the controller thread as well. If the input value is the chosen exit value, in this case the character 'p', the main thread calls interrupt on all other threads, waits for them to close, and lastly turns off the robot in a controlled manner.

Data Stream Thread. The data stream thread handles the data stream from the external tracker system. It is responsible for receiving and decoding the data into (x, y, z) coordinates and continuously updating the values in the data stream monitor.

Robot State Thread. The robot state thread queries the robot for its internal state 10 times per second and continuously updates the values in the robot state monitor.

Controller Thread. The controller thread is the thread that handles all of Spot's actions and movement from finding the fiducial on the tracker and orienting itself in the world, to moving Spot to new locations and position the marking device precisely using a feedback control loop. To command Spot, the control thread implements several high-level commands and helper functions for transforming points between different frames.

Data Streaming

Data are sent continuously from the tracker in a UDP (Leica) or TCP (Krypton) stream, depending on the tracker. It contains (x,y,z) coordinates and can also in-

clude timestamps and status information. The operator computer connects to the data stream through one of the threads within the bluelining program. The thread continuously receives the data, parses it and stores the latest coordinates in a monitor, protected by locks to avoid race conditions, where they are available to the control thread of the bluelining program upon request. The coordinate data are not filtered, but simply updated with the latest measured position.

Spot SDK

As mentioned in Section 3.1, Boston Dynamics provides an SDK for developing software to control and communicate with Spot.

The SDK provides several high-level commands that are useful for controlling and gathering data from Spot, and range from locomotion and manipulation commands to information queries and object registration.

Locomotion. The locomotion of Spot's body is obtained by providing position commands expressed in its *flat body* frame, which is a gravity aligned version of Spot's *body* frame, see Section 3.1. These commands allow Spot to move close to the desired point, but are not accurate enough to rely fully on.

Localization. The localization of the tracker is made by leveraging the SDK's World Object Service [Boston Dynamics, 2022j] for finding the tracker-mounted fiducial, providing Spot with a rough location of the tracker in the world. This allows Spot to keep the line of sight between the tracker and reflector by keeping its gripper pointed at the location of the fiducial while moving through the workspace.

Arm Control. The arm control utilizes several different methods depending on the situation.

The orientation sequence is achieved by providing positional commands to the Spot Arm expressed in the *flat body* frame. Since those commands are purely longitudinal, it allows for obtaining an estimate of the arm's position in Spot's local frame similar to if it was fixated in space even if the body balances, see Section 3.1. This results in a rather coherent estimate for the transformation between the global and the local frame when applying the Kabsch algorithm as covered in Section 2.1.

The feedback control sequence leverages velocity commands provided to the Spot Arm expressed in the *flat body* frame. The reason for controlling velocity instead of position in this situation is that velocity control allows for finer control of The Spot Arm. Any movement of the *body* frame during position control of the gripper specified in the *flat body* frame will translate and rotate the gripper's position according to the transformation between the *flat body* frame and the gripper's frame, see Section 2.1. If velocity control is used instead, the velocity direction will rotate according to the same transformation, but the translation part will have no effect.

To lock the arm for placing a mark when the point is reached, all arm joints are locked by explicitly setting their velocities to zero. The locking step is a compromise

that was needed because of the low sampling rate of the Leica AT403 combined with the stationary drift of the arm presented in the results. If a load disturbance occurs after locking, it will not be compensated for by the controller unless the operator decides to redo the fine positioning.

Queries. Spot is queried for its internal state in regular intervals, returning a Protobuf message with everything from joint states to saved world objects. The information from this query is used for obtaining the global position of the fiducial on the tracker and for ensuring that the marking device has contact with the ground by utilizing the force estimation the Spot Arm provides.

Control System

The developed control system incorporates positional control coupled with pressure control for ensuring the marking device has contact with the ground when placing the mark. There is also an optional offset compensation for countering the tilt problems that arise with a higher reflector position. The control loop runs at 5 Hz.

Positional Control. The positional control is handled by a PI controller, see Section 2.2, that takes the positional data from the data stream monitor as input, calculates the error relative to the current goal point, feeds it through the controller and outputs a control signal in the form of a velocity vector for the gripper, expressed in Spot's *flat body* frame. The integral part is needed to correct the stationary error that can arise from an inexact initially computed transformation between the global frame and Spot's *body* frame, and to compensate for Spot's active balancing, see Section 3.1, that causes the actual transformation between the frames to differ slightly from the computed one. To avoid an integrator windup, as described in [Wittenmark et al., 2021], a simple anti-windup scheme that caps the integral part to a maximum value is implemented.

Pressure Control. The pressure control for ensuring contact between the marking device and the ground is handled by a P controller that takes the estimated force on the end effector as input, calculates the error relative to the chosen set point which equals a force of 2 N on the ground, feeds it through the controller and outputs a control signal in the form of a vertical velocity that is sent to Spot together with the positional control command. In the case of an angled or vertical target surface, e.g., a wall, no changes to the control logic should be necessary since the gripper's frame follows the gripper's orientation and the change in gravitational load shouldn't affect the estimated force in the working direction noticeably. If a mark was to be made with the gripper rotated upside down, however, the gravitational load would have to be compensated for.

Offset Compensation. As mentioned in Section 3.2, there is no guarantee that the tracker's frame is perfectly aligned with the ground plane. On the contrary, it will never be perfectly aligned, since perfectly flat and horizontal floors do not exist.

To compensate for angular differences between the tracker's frame and the ground plane, an optional offset compensation method was developed that accounts for the ground plane's inclination relative to the tracker's frame. The compensation consists of a measuring sequence where three points on the ground plane around the goal point are measured and the inclination in both axes are computed. From this, the offset in both axes can be computed and compensated for.

For an even more robust estimation, a least squares estimation using a larger number of measured points could be made, which in addition would provide an estimation of the variance of the plane.

3.5 Full System Design

The full system can be seen in Figure 3.9 and consists of the following parts:

- Spot holding the marking assembly, with the reflector or LED mounted in the front, in its gripper.
- The tracker with its connected computer running some tracking software, streaming the positional data over a wireless network. The tracking software used at MAX IV is called Spatial Analyzer [New River Kinematics Metrology Institute, 2020].
- An operator computer connected to both Spot and the wireless network, receiving the streamed positional data and running the bluelining program, controlling Spot's movements.

In Figure 3.10 Spot and the Leica AT403 can be seen in the actual working environment at MAX IV.

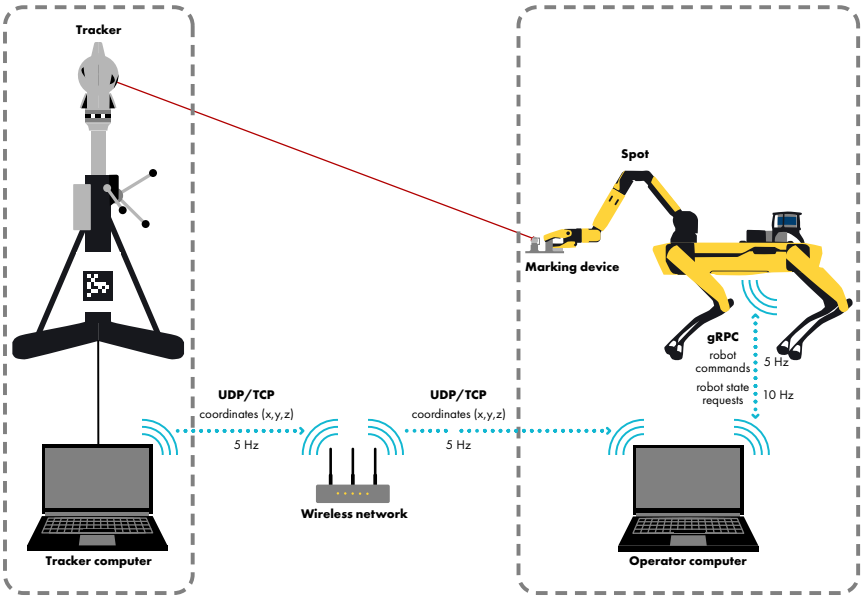


Figure 3.9 Schematic showing the full system design with all components.

Figure 3.10 Spot and the tracker in the actual working environment at MAX IV.

4

Spot Bluelining Workflow

This chapter presents the developed workflow used for bluelining an arbitrary amount of targets with Spot and describes each step.

4.1 Initialization of the Task

The initialization of the task consists of initiating the data stream, which will provide `NaN` values as long as there is no tracking, and positioning Spot where it is close enough to the tracker to register the fiducial mounted on the tracker. The maximum distance Spot can detect the fiducial from is dependent on the light conditions but 2 meters is usually enough. When Spot has found the fiducial, and in extension the tracker, it turns towards it and backs up to get in a better working range for the tracker. This distance is tracker specific, and in the case of the AT403, a distance between Spot and the tracker of 3 meters was deemed sufficient. Spot then deploys its arm and moves it to a default position slightly in front of its body. This step can be seen in Figure 4.1.

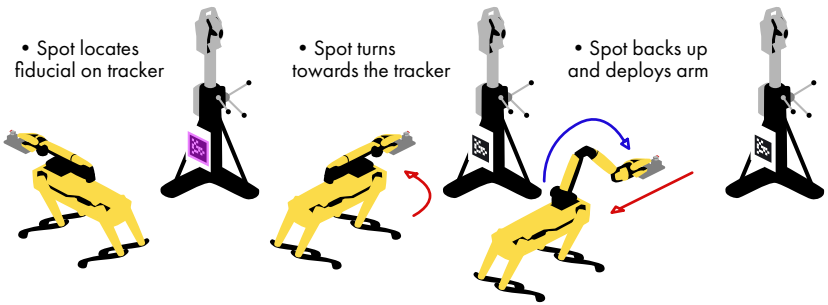


Figure 4.1 The initialization step, where Spot locates the fiducial on the tracker, turns towards it, and backs up and deploys its arm.

4.2 Orientation in the Tracker's Frame

When the tracker has found the reflector or LED on the marking device (manually or automatically depending on the tracker) and tracking is established, the data stream will start to provide positional data in the form of x (transversal), y (vertical) and z (longitudinal) coordinates. Spot then moves its arm in a straight line between two set positions where a static measurement is made at each position. The rotation between Spot's position and orientation and the tracker's frame is obtained from the measurements, by applying the Kabsch algorithm covered in Section 2.1, providing Spot with a reference to the tracker's frame. This step can be seen in Figure 4.2.

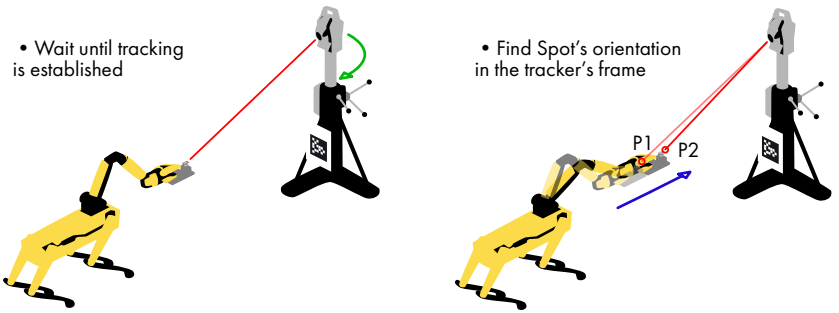


Figure 4.2 The orientation step, where Spot waits until tracking is established, and then finds its orientation in the tracker's frame by moving its arm between two set positions while taking static measurements, and then applying the Kabsch algorithm.

4.3 Positioning

The positioning task is divided into three steps. First, a coarse positioning step locating Spot's gripper roughly at the right coordinate is carried out using Spot's own sensors and features while allowing body locomotion. After the goal point is within reach, Spot calculates the ground plane's local offset relative to the tracker's frame to be able to compensate for it. Lastly, a fine positioning step is carried out. In this step, Spot keeps its body still and only moves the arm with closed-loop feedback from the tracker until the goal point is reached. Due to having the reflector and pen on the same axis, and therefore a fixed transformation between them, Spot does not need to grip the marking device exactly the same every time, as long as the grip is rigid enough.

Body Positioning

Spot's arm is moved to a position where it has the most reach along all directions, after which, the reflector position in the tracker's frame is measured. The goal point is then transformed from the tracker's frame to Spot's *body* frame and Spot is then commanded to move the body to a position where the reflector reaches the goal point. While moving to the goal point, Spot continuously keeps the reflector pointed at the tracker, keeping the line of sight. When Spot has reached the commanded point in its internal frame, its body is rotated towards the tracker and the error between the desired goal point and the actual position of the reflector is measured. If this error is within the reach envelope, specified as a circle around the goal point with a radius of 150 mm, Spot's body is lowered to the ground to increase the range of motion of the arm and the offset compensation is carried out. If the error is outside of this margin, the procedure is repeated. This step can be seen in Figure 4.3.

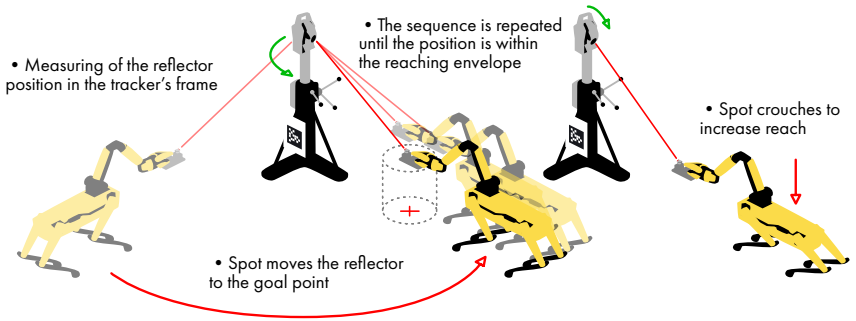


Figure 4.3 The coarse positioning step, where the reflector's position in the tracker's frame is measured and Spot is commanded to move the reflector to the goal point, repeating the sequence until the reflector is within the reach envelope. Spot then lowers its body to the ground to increase reach.

Line of Sight

The line of sight between tracker and reflector is kept during operation by commanding Spot to point its gripper at the tracker mounted fiducial. This is done by issuing a gaze command where the target is the saved position of the fiducial expressed in Spot's *vision* frame, and does not require the fiducial to be visible from Spot's current position. If the line of sight is broken, Spot will wait until it is established again before executing the next step. The current solution is for the operator to re-establish it manually, after which Spot continues from where the line of sight was lost.

Offset Compensation

During this step, Spot's body is kept stationary while the arm is moved in a triangular sequence around the goal point, taking static measurements of the reflector position at the vertices, while pressed firmly to the ground. These measurements are then used to compute the ground plane's local offset relative to the tracker's frame. This step can be seen in Figure 4.4.

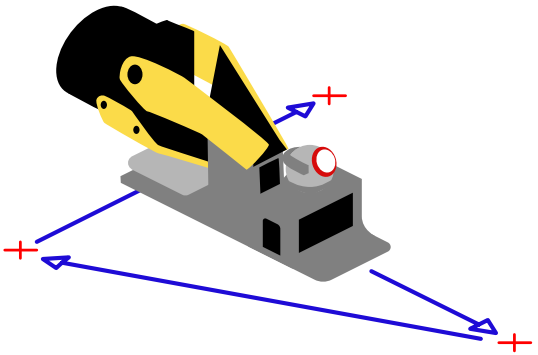


Figure 4.4 The offset compensation step, where Spot moves its arm in a triangular sequence around the goal point while pressing firmly against the ground and taking static measurements, which are then used to compute the ground plane's local offset relative to the tracker's frame.

Fine Positioning

Taking the calculated offset into account, Spot's arm moves the reflector towards the goal point while pressing lightly against the ground. Closed-loop PI control helps bring the reflector to a position within a radius of 1 mm from the goal point before locking the arm in place. After locking the arm, a static measurement is taken and if it is within the allowed margin, the operator can actuate the marking device. This step can be seen in Figure 4.5. If the measurement is outside of the allowed margin, the procedure is repeated.

4.4 Marking

When Spot has reached a point within an acceptable error margin of the goal point, the solenoids in the marking assembly are actuated by the operator with the use of an IR remote, as seen in Figure 4.5. This causes the pen tip to press down on the floor and make a mark.

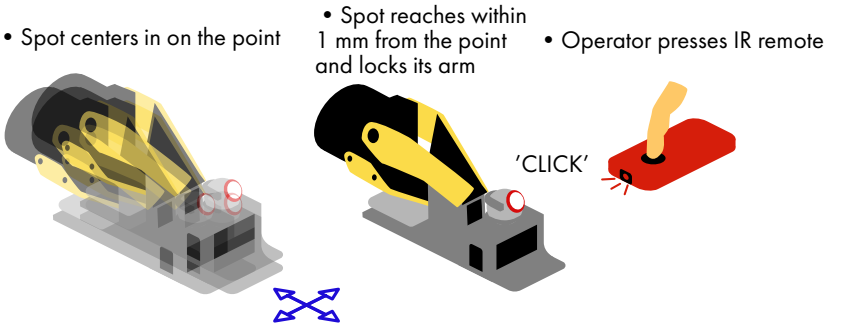


Figure 4.5 The fine positioning and marking steps, where Spot centers in on the goal point while pressing the assembly lightly against the ground, using closed-loop PI control to get within 1 mm from the goal point before locking the arm joints. If the position after locking is within the allowed margin, the operator actuates the pen with an IR remote, otherwise the sequence is repeated.

4.5 Flowchart of Workflow

In Figure 4.6, a flowchart detailing every step of the workflow can be seen.

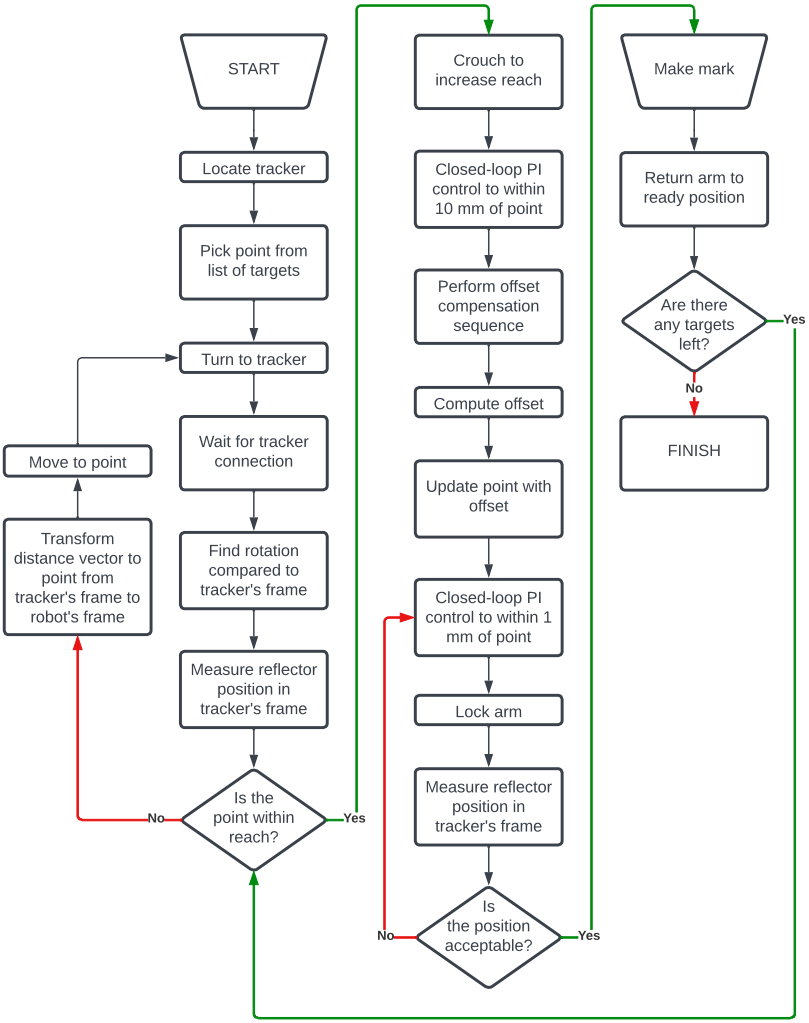


Figure 4.6 Flowchart of the proposed workflow for an arbitrary number of goal points.

5

Results

To assess the performance of the different developed subsystems and proposed solutions during development, the following experiments were carried out:

- To decide which frame to control Spot's body in, it was tested how much the different available frames drifted over time.
- To find how still the gripper could be held in space, an experiment of the arm drift was carried out with and without explicitly locking the arm.
- To choose which frame to move the Spot Arm in when doing the fine adjustment with tracker feedback, a test was carried out to explore how linear and orthogonal the arm moved when commanded in the *flat body*, *vision* and *odometry* frames.
- The marking device was tested separately, without using Spot, to gather data on the accuracy of the device itself compared to the conventional bluelining method.
- The offset compensation was tested by comparing points marked with it enabled to points marked with it disabled.

The results from these experiments are presented in Section 5.1.

When the system was finalized, the total system performance was evaluated by testing the accuracy, speed and adaptability. The results for these experiments are presented in Section 5.2.

5.1 Subsystem Tests

Spot

Body. To decide which one of the *odometry* and *vision* frames to control Spot's body in, it was tested how much the two different frames drifted over time. The

test was made by first having Spot lie down and continuously measure the relation between these respective two frames and the *body* frame with one measurement per second for five minutes. Thereafter, the same kind of test was performed with Spot standing up at the same place for five minutes.

In Figures 5.1 and 5.2 the results of these two tests can be seen. It is clear that the *odometry* frame fluctuates more than the *vision* frame over time. While the vision frame shows a slight error along both axes while standing, this error is mostly static over time, though rather noisy with sharp changes in the positional estimate. The positional estimate is less noisy in the *odometry* frame than in the *vision* frame, and while the mean error from nominal for the position in the x-direction when standing is smaller in magnitude in the *odometry* frame than in the *vision* frame, it fluctuates wildly. When sitting down, the error in the *odometry* frame increases over time, especially in the y-direction.

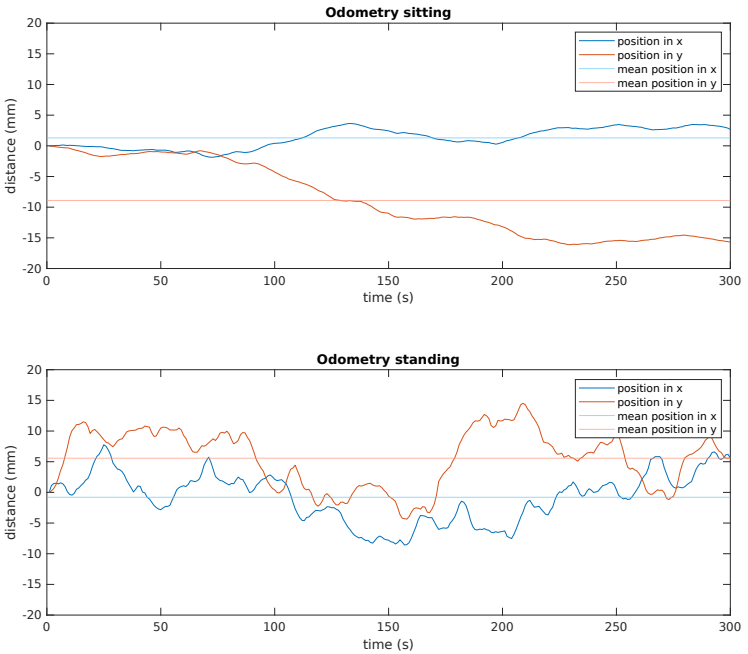


Figure 5.1 Test results, showing how the *odometry* frame drifts over time when Spot is sitting down, and fluctuates when Spot is standing up, with no commanded motion during the experiments.

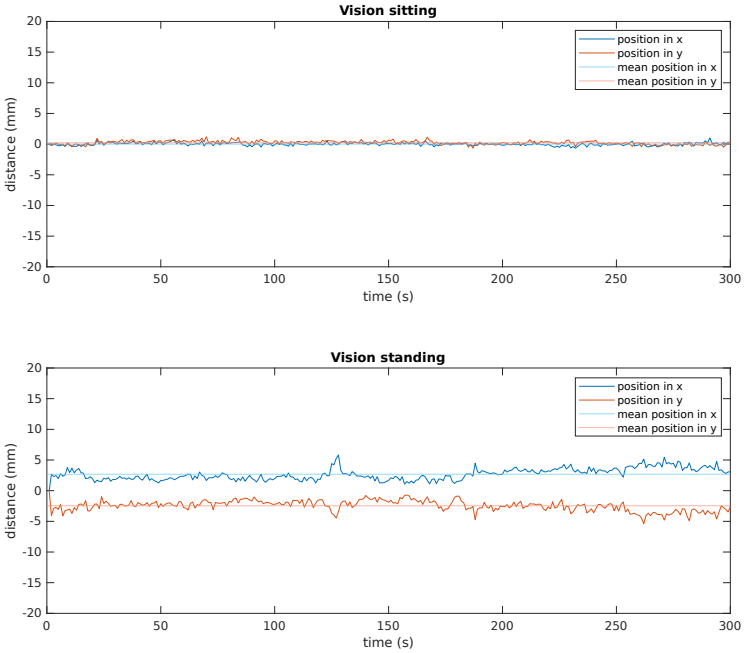


Figure 5.2 Test results, showing how the *vision* frame fluctuates over time when Spot is sitting down, and standing up, respectively, with no commanded motion during the experiments.

Arm. When conducting the first tracking tests, it quickly became clear that the arm was drifting when no commands were issued. Drifting makes it harder to characterize and model the system and decreases the ability to both control it and have it stay at the same position a longer period of time. The problem was mitigated by locking the arm in position by explicitly setting all joint velocities to zero resulting in the arm drifting less over time by an order of magnitude. The difference in drift between not locking and locking the arm over the period of one minute can be seen in Figure 5.3.

To choose which frame to move the Spot Arm in, a test was carried out to explore how linear and orthogonal the arm moved when commanded in the *flat body*, *vision* and *odometry* frames. Spot was lined up with the tracker's frame while holding a device containing an LED in its gripper and was commanded to move it repeatedly in a cross shape purely along the x- and y-directions of the respective frames, returning to origin between every round. The result can be seen in Figure 5.4. A visual assessment shows that while all tests show reasonable orthogonality, the *odom-*

etry frame is heavily rotated compared to the tracker frame, and in extension Spot's *body* frame. The *flat body* and *vision* frames have similar rotations, lined up with the tracker's frame, and the *flat body* frame shows the least deviations between rounds.

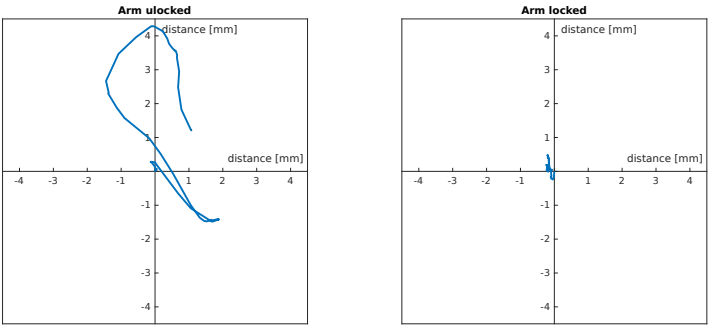


Figure 5.3 Test results of arm drift over one minute with the arm unlocked, and locked, respectively.

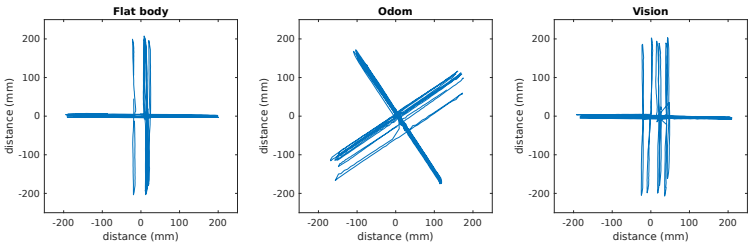


Figure 5.4 Test results of the arm's orthogonality and linearity in movements in the *flat body*, *odometry*, and *vision* frames, respectively.

Marking Device

When the first prototype of the pull-down marking device was finished, it was tested at MAX IV. Initially, the marking device was tested separately, without using Spot, to gather data on the accuracy of the device itself compared to the conventional bluelining method.

The test was made by choosing 12 arbitrary points in the Leica AT403 coordinate system and making a marking at each position with the marking device. After marking the points the conventional bluelining tool was positioned right above the points and the position of the marking tool was then measured with the Leica laser tracker. This provided data on the offset and accuracy between the marking device

and the conventional bluelining tool. In Figure 5.5, the relative distance from ground truth, which in this case is the bluelining tool, is presented and it can be seen that a rather even spread around ground truth of less than 300 µm was achieved in most cases, with a mean radius of 170 µm.

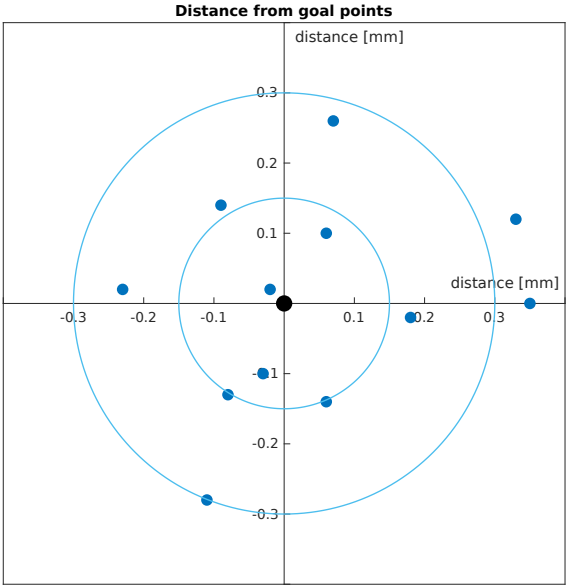
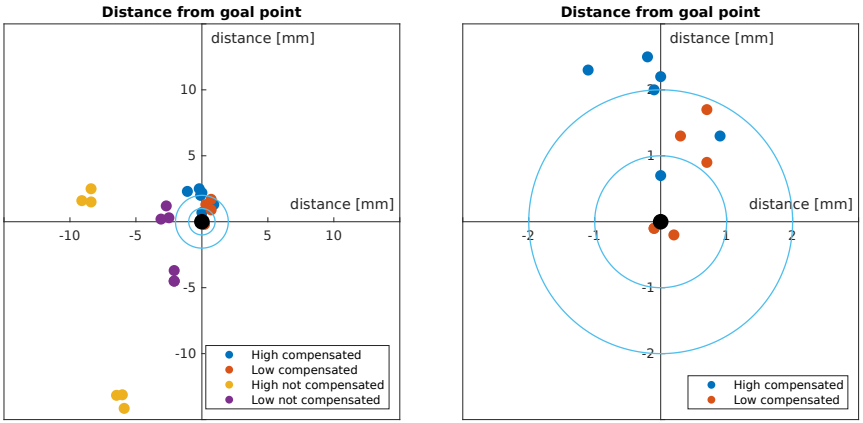


Figure 5.5 Accuracy test results for the isolated marking device for 12 arbitrarily chosen points.

Offset Compensation

The offset compensation was tested with the Krypton K610 by marking a point repeatedly, with the compensation turned on and off, respectively. To ensure misalignment of the tracker frame and the ground plane, the tracker was tilted between 5° and -5° around both horizontal axes. The result can be seen in Figure 5.6, with all of the marked points visible in Figure 5.6a and only the compensated points visible in Figure 5.6b.



(a) Zoomed out plot showing all marked points.

(b) Zoomed in plot showing the compensated points.

Figure 5.6 Test results for the offset compensation, with different reflector positions.

Table 5.1 The mean offsets for the compensated points and the radii for the different set ups in the offset compensation experiment.

Set up	Comp on			Comp off
	Mean z-offset (mm)	Mean x-offset (mm)	Mean radius (mm)	Mean radius (mm)
High position	1.8	0.4	1.9	10.6
Low position	0.7	0.3	0.8	3.4

It can be seen that the compensation provides results mostly within the allowed 2 mm radius, but there seems to be a tendency to place points a little forward in the tracker's z-axis (the vertical axis in the plots). This tendency gets more pronounced with the higher reflector position. The mean radii from the target point for the different set ups are presented in Table 5.1, together with the offsets in both z- and x-directions.

5.2 Full System Tests

Marking Accuracy

The marking accuracy for the full system was tested by marking five points, within a 3×3 m^2 area, at the actual proposed workspace, i.e., the concrete floor of MAX

IV's large storage ring facility, and a short video of this process can be seen in [Gulz-Haake and Karlbrink Malmquist, 2023a]. The points were grouped in two groups, consisting of two and three points, respectively, located in each corner of the workspace, simulating a possible actual bluelining task. Spot was placed so it could find the tracker mounted fiducial, and apart from the initial establishing of the line of sight between reflector and tracker, no manual intervention was carried out. The experiment was run four times, with both a low and high reflector position, and the points were afterwards measured with the manual bluelining tool and the deviation was calculated. The results from the final test at MAX IV can be seen in Figure 5.7.

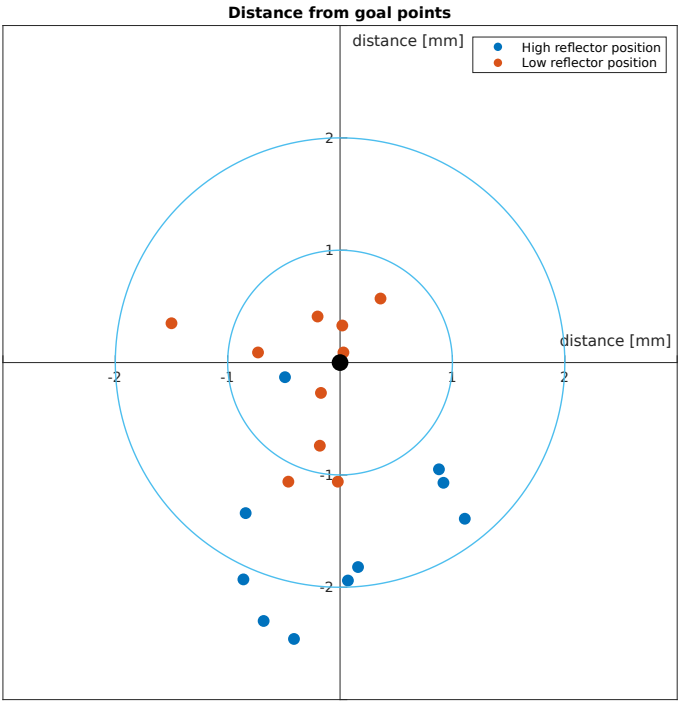


Figure 5.7 Test results showing distance to nominal points for different reflector positions over four tests, with five goal points each. The results show an apparent systematic bias, consistently placing the points made with the high reflector position behind the ones made with the lower reflector position.

First and foremost, it can be seen that most of the measured points are within a 2 mm radius from the target point. Furthermore, it can be seen that the error often is larger for the points with a high reflector position, than for those with a lower, and that there seems to be a systematic bias for the points with the higher reflector position, positioning them between 1 and 2 mm behind the target.

The mean radii to the target points for the two positions can be seen in Table 5.2 and are 1.7 mm for the high position and 0.6 mm for the low position.

When looking at the points made with the lower reflector position, it can also be seen in Figure 5.7 that in 73 % of all cases, the points are within 1 mm of the target position, whereas with the higher reflector position only 9 % of the points are within 1 mm of the target.

Table 5.2 The mean offsets and radii for the different set ups in the final experiment.

Set up	Mean z-offset (mm)	Mean x-offset (mm)	Mean radius (mm)
High position	1.5	0.6	1.7
Low position	0.5	0.4	0.6

Process Times

When conducting the final experiments at MAX IV, the fine-adjustment process was logged for every point. The result from one of the runs can be seen in Figure 5.8. It can be seen that the fine-adjustment step always starts from the last measured position during the offset compensation and repeatedly closes in on the goal point in about 5 seconds. After reaching a position within millimeters of the goal point, the time until the mark is placed differs depending on how many iterations are required to be done before the arm locks in a satisfactory position. The average time from finishing the offset compensation to marking a goal point is found to be around 30 seconds. In this test case, that includes time for measuring the reflector position for later validation.

Adding to the total process time, the following process times were estimated from analysis of video material from the experiments:

- The offset compensation sequence has a static duration of 24 seconds.
- The initial boot up and orientation sequence, including turning the tracker, takes around 25 seconds.
- The coarse localization step for each point differs, but was found to average to around 25 seconds.

Adding these up, the total process time comes out to around 1 minute and 30 seconds per point.

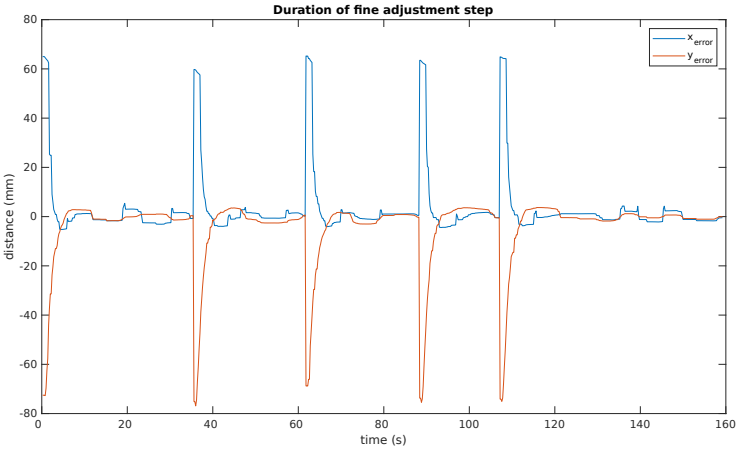


Figure 5.8 Plot showing the fine-adjustment step for each point during an experiment at MAX IV. From the horizontal axis, the process time for each point can be derived by looking at the time elapsed between each peak, which denote the start of a fine-adjustment step.

6

Discussion

The results in Chapter 5 show that it is indeed feasible to use a robot like Spot to perform the bluelining task. The results clearly indicate that Spot itself is not accurate enough, neither in its positional estimation or joint estimates, to perform the task. However, using continuous closed-loop feedback from an external tracking system and a simple control algorithm, it is clear that these inaccuracies can be compensated for, in order to achieve the desired accuracy.

The development process showed that it is possible to design a general solution for different tracking systems, by decoupling the positional data stream from the remaining parts of the program. By doing this, all system logic can be kept while changing tracking system, and when introducing a new system, only the data stream handling has to be implemented for that new system. However, consideration has to be taken that different measurement systems might need different parameter tuning.

To account for differences in measurement systems when introducing new systems, or changing between systems, some kind of automatic tuning could be implemented. However, that was outside of the scope of this thesis project.

6.1 Performance

Spot Performance and Limitations

The results shown in Figures 5.1 and 5.2 indicate that Spot's own positional estimation lacks the accuracy for a high accuracy task such as bluelining. It is, however, good enough for iterative coarse positioning with the support from periodic control measurements as shown by the full system results in Section 5.2.

During development, different sizes for the reaching envelope were tested, and it was found that anything smaller than a 100 mm radius caused serious problems for Spot, often resulting in an endless adjustment sequence where Spot never reached a position within the envelope.

Spot's arm has a far higher positional resolution than the rest of the robot, but still suffers from inexact positional estimation and joint estimation as shown by the results in Figure 5.4. The arm drift, seen in Figure 5.3, can likely be explained by

Spot trying to keep its arm still in the *vision* frame. When this frame fluctuates, the arm follows the fluctuations, causing the drift.

All of this limits Spot in high precision and accuracy operations, but the results also show that when providing Spot with a means to relate to an external fixed coordinate system, these issues can be worked around.

While it was found that is was possible to program Spot to retain all high-level default autonomous features while restraining some of its movements, the lack of low level control and documentation, something we named "the black box problem", sometimes complicated things. For example, it showed to be really hard to keep Spot's gripper pointed at the reflector, while at the same time having the arm following the moving body. It also showed to be hard to keep Spot in a crouched position while simultaneously pressing the arm to the ground with the use of pressure control.

When programming the behaviors described above, it became clear that the simpler the command could be written, e.g., only addressing one feature at a time and keeping the parameter count for each command low, the higher the probability that it would work as intended. A guess might be that simple commands allow Spot's autonomous features to work more uninterrupted by conflicting logic, but without having access to Spot's source code, no definite conclusion can be made.

One final finding is that the covering of the front facing depth cameras and the gripper's depth camera did not seem to affect the autonomous performance.

Full System Performance

As seen in Figure 5.7, the system with the low reflector position achieved an accuracy of 1 mm or less in 73 % of all test cases, and an accuracy within 2 mm in 100 % of all test cases. With the high position, the accuracy within 2 mm dropped to 70 %, with only one point placed within 1 mm. This, together with the results from the isolated offset compensation experiments shown in Figure 5.6 indicates that there is a constant bias when using the higher reflector position. This bias can have several causes. The higher position extends the distance between reflector and pen tip and increases the potential for tilt problems. These can be caused both by physical misalignment of the reflector and pen tip, and by misalignment between the floor and the tracker's frame. If the error is caused by physical misalignment, it could be countered by using tighter tolerances when constructing the marking device, or by compensating for it with a fixed offset. If it is instead caused by a noticeable discrepancy between the floor and the tracker's frame, that could mean that the proposed offset compensation sequence does not work fully as intended.

Noteworthy is that the results from Figure 5.6 and Figure 5.7 show constant biases in opposite directions. This indicates that there is some physical misalignment that could be caused by the plastic elastically deforming when assembling the device and then deforming differently when reassembling it, resulting in a change in bias. This makes it difficult to implement a static offset compensation, since it

would have to be changed every time the device is reassembled. An alternative solution could be to start every bluelining task with a reference point that has been precisely measured. Spot would mark the point and could thereafter use a camera to detect the points and measure the distance between them. This distance could then be used to calculate the static offset and compensate for it. A difficult aspect of this approach would be identifying and separating the correct points, especially at a floor like MAX IV which is almost speckled. It would also require manually measuring a reference point every session, or having a tracking system permanently stationed with a fixed reference point in its tracking range that could be used for this calibration.

Even with the bias, an accuracy within 2 mm from the goal point is enough to accommodate the required adjustment range for equipment installation which means the system did achieve the main technical objective when looking at the mean radius in Table 5.2. The trajectory planning was mainly left to Spot's internal system, while the developed code focused on ensuring proper orientation and keeping the line of sight.

During the final tests, the line of sight was never broken, but as stated in Section 4.3 there is a solution implemented that waits for the line of sight to be reestablished if it is lost.

Compared to the performance of a human operator performing a manual bluelining sequence, as presented in [Andersson and Shahin, 2022b], the performance of the developed solution is on par both regarding accuracy and speed judging by the experiment results.

6.2 Usability

While the results in Chapter 5 show us that it is feasible to use a robot like Spot for the bluelining task, they provide very little information regarding the usability of the developed solution.

Although working quite well, the current solution does require some getting used to, to be able to use it properly. First of all, the solution is in its current state entirely developed for executing from the command line meaning the user of the program will most likely need to have some sort of command line experience. Furthermore, all options and choices proposed to the user throughout the bluelining process, such as telling the program when it is time to move on to the next point is all done through the command line. A solution to this could be to replace the command line interface with a GUI. This would require less knowledge about the system and the command line and thus making the program easier to use for a broader range of people.

The current solution also requires the operator to actuate the solenoids with an IR remote to actually make the mark, which together with having to position Spot

close to the fiducial before starting the bluelining sequence reduces the level of autonomy significantly.

6.3 Further Development

To make the system truly autonomous, increase usability and extend it to other application areas, there are different aspects that need to be further explored and developed.

Marking Device

The marking device itself needs to be further developed. Instead of actuating the solenoids with an IR remote, a solution that allows the bluelining program to communicate with the device needs to be constructed. One solution, that has been looked into but not realized in this thesis is to create a custom gRPC service running on the device and to connect it as a communication endpoint via the electrical interface of Spot's gripper, which has both power supply and Ethernet capability. This would allow for writing a custom gRPC client that sends custom Protocol Buffer messages to the gripper, routed through Spot via port forwarding. This solution would require a mechanical design for the connector to the electrical interface that allows Spot to physically connect to the marking device when picking it up, and to disconnect when putting it down, to achieve full autonomous operation.

If the marking device is to be used on walls and other vertical surfaces, the reflector mounting would also need to be rethought. It is presently mounted at a fixed angle and kept in place magnetically, which works well on the floor, but does not ensure line of sight when moving on walls. One solution could be to use a motorized mount that tracks the laser tracker, another could be to use a reflector with a larger acceptance angle.

There is also a need to investigate other solutions for the actual mark making, as a permanent marker dries out when left exposed over time. It might be feasible to use a small inkjet printer head instead, which would also provide better control over point size.

Operation

Spot needs to be able to pick up the marking device by itself. This can be solved by placing the device in some kind of dock. The dock could have a fiducial on it, similar to the available self-charging dock [Boston Dynamics, 2022a] and a design that guides the gripper to a position that ensures an adequate grip of the handle. The grip does not have to be exactly the same every time. As long as the device is held firmly, the accuracy should not be noticeably affected since the reflector an pen tip are mounted on the same axis.

More redundancy also needs to be developed. If the line of sight between the tracker and Spot is broken, the current procedure is to wait for the operator to

reestablish it and then continue. For autonomous operation, a solution where Spot and the tracker search for each other needs to be implemented. Since Spot knows where the tracker is located and can make sure to look at it with fairly high accuracy, a simple search function where the tracker scans an area according to a smart algorithm (last seen, next point, etc) could be enough.

As for planning, the current solution simply takes a list of points and traverses them in the specified order. This works for a smaller set of points, but for larger tasks with many points to be marked, it would probably be beneficial to optimize the task. There are several ways to do this, but one automated solution could be to cluster the points and then finding the optimal way to traverse the clusters.

Lastly, the robustness needs to be further assessed and developed. The floor at MAX IV is very flat and clean, but even a small rough patch or pebble could disturb the fine positioning or offset compensation sequence. An extension of the offset compensation sequence could be to fit the floor plane to a larger set of measured points using least squares estimation, which in addition would provide an estimation of the variance of the plane. The software also needs to be able to handle unforeseen events. Extensive testing would have to be made to explore how different disturbances affect the system and how they can be mitigated.

Other Application Areas

The system proposed in this thesis would not be limited to the specific task of making marks on a concrete floor, but could be extended to other areas as well. This can, e.g., include similar processes at other research laboratories as well as marking and line drawings during construction of, e.g., fairs. When setting up a fair, the placement of booths, tables, screens, attractions etc. have to be marked on floors and walls. The precision required is usually in the range of centimeters rather than millimeters. Another area of interest is construction sites. During construction work, lines, marks, and other things have to be drawn at the construction site at the ground, floors, walls, etc. There is also a need for follow up and control measurements during the construction, to make sure the work goes as planned and to find deviations in time.

Some adaptations to the proposed system would have to be made for it to be suitable for applications in other environments. At construction sites, the marking device would likely have to be redesigned and more rugged, with a different marking method such as a nozzle. At both fairs and construction sites, there is often a greater need for writing annotations and drawing symbols, not just marking points. This could be solved by adapting the developed workflow to allow drawing with some kind of path following, or by implementing an inkjet printer head with a large enough printing area. The tracking might also have to be rethought, with several trackers to cover more space, or a different tracking method. The obstacle avoidance feature would also likely be more important in the other applications areas, especially those including a lot of movement around the robot such as at construc-

tion sites and during constructions of fairs and festivals. Preliminary tests show that the obstacle avoidance integrates well with the current system, but it would have to be further investigated to ensure proper functioning in all cases.

Spot already has data gathering capabilities and is used in the industry for surveillance work [Boston Dynamics and Swinerton, 2022]. In [Nilsson, 2021], a method for reconstructing construction sites with SLAM and Spot's LiDAR was investigated, and coupled with an adapted version of the system design proposed in this thesis, Spot comes closer to being a fully fledged foreman.

7

Conclusion

This thesis has presented a gripper-held marking device developed to enable Spot to perform a bluelining task, together with a full software solution. It has been shown that the required marking accuracy of 2 mm can be achieved repeatedly in the real working environment at MAX IV. The process time was found to be around 1 minute and 30 seconds per point, which is similar to a human operator for a single point. However, over time, with many points to mark this solution most likely starts to exceed human capacity. The actuation of the marking device is currently a limiting factor to the automation of the entire process, since it is decoupled from the software solution and relies on manual operation. Even with this limitation, the ergonomics for the operator would be noticeably improved compared to the fully manual process, since the time spent crouched on the floor would be reduced.

From the presented results regarding accuracy, and process time, it can be concluded that Spot has the ability to be maneuvered to mark points with high accuracy compared to previous solutions for automating the bluelining process. Using Spot and the proposed workflow results in faster process times while also being able to mark points over a bigger working area with high precision. Throughout this process, Spot can be adapted to a constrained movement scheme, while retaining its autonomous features like obstacle detection and balancing. The line of sight is kept throughout the task by always keeping Spot's gripper pointed at the tracker. If the line of sight is lost, Spot stops until it is manually re-established.

It has furthermore been concluded that it is feasible to use Spot equipped with the Spot Arm to perform high precision and accuracy operations. Nonetheless, there are limitations for using Spot for these kind of tasks. It could be seen from the subsystem tests that the Spot arm is drifting when no commands are issued, and although the problem was mitigated it can be concluded that there is a clear limitation to how still the Spot Arm can stay over time which in turn is a clear limitation for using it for high accuracy operations.

Two different reflector/LED heights were evaluated, and it was concluded that the further from the pen tip the reflector/LED is placed, the greater the risk becomes for angular errors caused by floor tilt, physical misalignment, etc. An angular offset

compensation was implemented showing promising results when tested on heavily tilted surfaces.

Lastly, from the results and data gathered during the thesis, it can be concluded that the proposed solution shows promise regarding both accuracy and autonomy. It can therefore be concluded that similar solutions have the potential to be expanded into other areas and automate similar processes, for example at other research laboratories, when constructing fairs, and at construction sites.

Bibliography

Andersson, A. and A. Shahin (2022a). "MAX IV high precision self-positioning robot". In: *International Workshops on Accelerator Alignment 2022* (CERN, Oct. 31–Nov. 4, 2022). IWAA. Geneve.

Andersson, A. and A. Shahin (2022b). *MAX IV high precision self-positioning robot*. https://indico.cern.ch/event/1136611/contributions/5020021/attachments/2528117/4370081/BlueliningrobotMaxIVIWAA2022.pdf. Accessed: 2023-01-09.

Åström, K. and T. Hägglund (1995). *PID Controllers: Theory, Design, and Tuning*. ISA - The Instrumentation, Systems and Automation Society. ISBN: 1-55617-516-7.

Boston Dynamics (2022a). *About the Spot Dock self-charging station*. https://support.bostondynamics.com/s/article/About-the-Spot-Dock-self-charging-station. Accessed: 2022-12-29.

Boston Dynamics (2022b). *About the Spot robot*. https://support.bostondynamics.com/s/article/About-the-Spot-robot. Accessed: 2022-12-29.

Boston Dynamics (2022c). *Boston Dynamics Networking*. https://dev.bostondynamics.com/docs/concepts/networking. Accessed: 2022-12-28.

Boston Dynamics (2022d). *Boston Dynamics Payload Software Interfacing*. https://dev.bostondynamics.com/docs/payload/configuring_payload_software. Accessed: 2022-12-28.

Boston Dynamics (2022e). *Boston Dynamics Spot SDK*. https://github.com/boston-dynamics/spot-sdk. Accessed: 2022-12-31.

Boston Dynamics (2022f). *Boston Dynamics Spot SDK*. https://dev.bostondynamics.com/docs/concepts/geometry_and_frames. Accessed: 2022-09-30.

Boston Dynamics (2022g). *Boston Dynamics Spot SDK Concepts*. https://github.com/boston-dynamics/spot-sdk/blob/master/docs/concepts/README.md. Accessed: 2022-12-31.

Boston Dynamics (2022h). *Spot Arm specifications and concepts*. https://support.bostondynamics.com/s/article/Spot-Arm-specifications-and-concepts. Accessed: 2022-12-29.

Boston Dynamics (2022i). *Spot Enhanced Autonomy Package*. https://support.bostondynamics.com/s/article/Spot-Enhanced-Autonomy-Package-EAP. Accessed: 2022-12-29.

Boston Dynamics (2022j). *World Object Service*. https://dev.bostondynamics.com/python/bosdyn-client/src/bosdyn/client/world_object. Accessed: 2022-12-27.

Boston Dynamics and Swinerton (2022). *Legged Automation: The final piece in end-to-end construction progress tracking*. https://www.bostondynamics.com/resources/case-study/swinerton. Accessed: 2022-12-29.

Cognibotics AB (2023). *Cognibotics*. https://cognibotics.com/. Accessed: 2023-01-17.

Frida Nilsson/MAX IV (2023). *MAX IV Schematic*. Illustration. MAX IV, Lund University.

Goetz, B., T. Peierls, J. Bloch, J. Bowbeer, D. Holmes, and D. Lea (2006). *Java Concurrency in Practice*. ISBN-10: 0321349601 ISBN-13: 978-0321349606. Addison-Wesley, Boston, MA.

Google (2023). *Protocol Buffers*. https://protobuf.dev/overview/. Accessed: 2023-02-25.

Gulz-Haake, S. and N. Karlbrink Malmquist (2023a). *Bluelining at MAXIV*. Accessed: 2023-01-18. Youtube. URL: https://youtu.be/hmhdsrzPs0Q.

Gulz-Haake, S. and N. Karlbrink Malmquist (2023b). *Spot Bluelining Repository*. https://github.com/sebastiangulzhaake/spot-bluelining. Accessed: 2023-03-13.

Hägglund, T. (2019). *Lecture Notes for Automatic Control*. Lund University, Department of Automatic Control.

Inoue, F. and E. Ohmoto (2012). "Development of high accuracy position marking system in construction site applying automated mark robot". In: *Proceedings of SICE Annual Conference*. SICE, pp. 819–823.

Kabsch, W. (1976). "A solution for the best rotation to relate two sets of vectors". *Acta Crystallographica Section A* **32**, pp. 922–923.

Klinghav, L. (2021). *Mobile Floor-Marking Robot, utilizing Feedback from Laser Tracker*. Lund University, Department of Automatic Control. MSc Thesis TFRT-6123. Available for download at: https://lup.lub.lu.se/student-papers/search/publication/9041939.

Krypton Electronic Engineering (2003). *Krypton Help Pages on K400/K600 Hardware & Software Guide*.

Leica Geosystems (2017). *Leica AT403 User Manual*.

Lippitsch, A. (2018). "Leica Absolut Laser Tracker Operation in Magnetic Field Environment". In: *International Workshops on Accelerator Alignment 2018* (Fermilab, Oct. 8–12, 2018). IWAA. Batavia, IL.

Lund University, Faculty of Engineering (2023). *Center for Construction Robotics*. https://www.lth.se/digitalth/byggrobotik/. Accessed: 2023-01-02.

Lund University, MAX IV (2023a). *MAX IV*. https://www.maxiv.lu.se/. Accessed: 2023-01-02.

Lund University, MAX IV (2023b). *MAX IV Beamlines*. https://www.maxiv.lu.se/beamlines-accelerators/beamlines/. Accessed: 2023-01-02.

Lund University, MAX IV (2023c). *MAX IV Facility*. https://www.maxiv.lu.se/about-us/the-max-iv-facility/. Accessed: 2023-01-02.

Lynch, K. and F. Park (2017). *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press. ISBN: 978-1107156302.

New River Kinematics Metrology Institute (2020). *SpatialAnalyzer User Manual*.

Nilsson, O. (2021). *Building dense reconstructions with SLAM and Spot*. Lund University, Department of Automatic Control. MSc Thesis TFRT-6158. Available for download at: https://lup.lub.lu.se/student-papers/search/publication/9079252.

Patil, V. V. (2022). *High Precision Robotic Manipulator for Bluelining at MAX IV*. Lund University, Department of Automatic Control. MSc Thesis TFRT-6173. Available for download at: https://lup.lub.lu.se/student-papers/search/publication/9095103.

Python Software Foundation (2022a). *Python Multiprocessing Documentation*. https://docs.python.org/3/library/multiprocessing.html. Accessed: 2022-12-28.

Python Software Foundation (2022b). *Python Threading Documentation*. https://docs.python.org/3/library/threading.html. Accessed: 2022-12-28.

Python Software Foundation (2023). *Python Global Interpreter Lock (GIL) Documentation*. https://docs.python.org/3/glossary.html\#term-global-interpreter-lock. Accessed: 2023-01-02.

Tsuruta, T., K. Miura, and M. Miyaguchi (2019). "Mobile robot for marking free access floors at construction sites". *Automation in Construction* **107**, p. 102912.

Wittenmark, B., K. J. Åström, and K.-E. Årzen (2021). *Computer Control: An Overview*. E-husets tryckeri, Lund.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER'S THESIS

Date of issue
February 2023

Document Number
TFRT-6189

Author(s)
Sebastian Gulz-Haake
Nils Karlbrink Malmquist

Supervisor
Alina Andersson, MAX IV Laboratory, Lund University, Sweden
Mathias Haage, Cognibotics , Sweden
Björn Olofsson, Dept. of Automatic Control, Lund University, Sweden
Anders Robertsson, Dept. of Automatic Control, Lund University, Sweden (examiner)
Bo Bernhardsson, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle
Automation of High-accuracy Marking Tasks at MAX IV using the Quadrupedal Robot Spot

Abstract

When installing new research equipment at the synchrotron research laboratory MAX IV, a final positional accuracy of tens of microns is usually needed. To obtain that accuracy, initial reference points must be placed within 2 mm from the nominal target points in CAD, in a process referred to as bluelining. It is currently a manual process where points from a virtual environment are transferred to the real working environment and marked with the help of a laser tracker.

The purpose of this Master's Thesis is to explore the possibilities of automating the bluelining process using a 12 DOF quadrupedal robot equipped with a 6 DOF and gripper robotic arm together with a Leica AT403 external tracking system. The specific robot is the robot dog Spot from Boston Dynamics equipped with a Spot Arm. The system that has been developed consists of a platform carried by Spot's gripper containing the tracked reflector and a marking assembly, and of a software solution controlling the robot.

The proposed workflow starts with a localization step where Spot finds the tracker and orients itself in the tracker's frame. Once oriented, Spot positions itself within reach of a target point before positioning the marking device on the point using closed-loop feedback control. Once the point is reached, the operator actuates the marking device remotely after which Spot continues to the next point.

With this system, a marking accuracy within 2 mm from the target was achieved repeatedly in the real working environment at MAX IV. The process time was found to be around 1 minute and 30 seconds per point, which is comparable to a human operator.

From the results and data gathered during the project, it can be concluded that the proposed solution shows promise regarding both accuracy and autonomy. Furthermore, it can be concluded that similar solutions have the potential to be expanded into other areas and automate similar processes, such as marking processes for equipment installation at other types of research laboratories, as well as for booth setups at exhibitions and fares, and at construction sites when marking out mounting locations.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title		ISBN
0280-5316		
Language	*Number of pages*	*Recipient's notes*
English	1-49	
Security classification		

http://www.control.lth.se/publications/