Project Thesis

The automated welding industry;
how it has been achieved and the challenges it faces

Thea Holmedal

2020-12-16

Preface

This report is written in accordance with a mandatory course for students at The Department of Mechanical and Industrial Engineering, completing their degree within Robotics and Automation at NTNU. The course permits the student to chose a field in which to specialize with the guidance of a supervisor.

I want to express my gratitude to supervisor Lars Tingelstad for his guidance throughout the semester. In addition, I want to thank Vebjørn Bergsholm Bjørhovde for a valuable collaboration with both interviews conducted for this report as well as discussions regarding our field of specialization. Also, I would like to thank Adam Leon Kleppe for providing guidance and useful information regarding the robot at disposal for this project. Lastly, I would like to give thanks to the welding robot personnel interviewed for their time and indispensable knowledge.

The content of this report is based on literature, where the rightful owner is listed. Further, the interviews conducted were both prepared and executed in collaboration with Vebjørn Bergsholm Bjørhovde.

Summary

This report is written to provide the reader with a general understanding of robotic welding with its possibilities and limitations through a "state of the art" review of the industry. The robotic welding industry has evolved substantially since the invention of the first industrial robot. This report presents different aspects that make the automated welding industry possible.

While there are many reasons for automating the welding process, such as increase in accuracy, higher production rate, and reduction in human errors, the automated welding industry still faces some challenges. The main problem lies within the fact that, despite incredible advances within the robotic industry, a robot is still not fully able to resemble a human. Not only does a robot require a detailed description of its task to act, but it also requires sensors in order to react.

The description of a task is done by programming the robot, where different methods exist, with their advantages and disadvantages. The implementation of sensors allows for the robot to react, but this technology faces challenges due to the harsh environment accompanying any welding operation. Further, a welding robot requires the implementation of welding equipment, where the automated welding equipment differs from the ones utilized in manual welding.

One topic of research today is constraint-based programming of robots. This method aims at making a robot system more adaptable to ever-changing environments and thereby allowing for implementation in new segments of the industry.

Acronyms

CMT Cold Metal Transfer. 27, 41

DOF Degrees of freedom. 10

eTaSL Expressiongraph-based Task Specification Language. 32, 33

eTC Expressiongraph-based Task Controller. 33

FSW Friction Stir Welding. 27

GMAW Gas Metal Arc Welding. 24, 27

GTAW Gas Tungsten Arc Welding. 24

OLP Offline programming. 31

PAW Robotic Plasma Arc Welding. 26

PoE Product of Exponentials. 12

VS-code Visual Studio Code. 37, 39, 40

Contents

List of Figures

List of Tables

Chapter 1.

Introduction

The subject of this report is robotic welding. The reader will be introduced to the industry of automated welding, from its origin till today, including different aspects contributing to making automated welding possible. Further, a system description of the welding cell at disposal for this project is given to provide context to some of the concepts discussed in this report.

1.1. Objectives

This report has three main objectives, which are listed below:

- Present a state of the art review of the robotic welding industry
- Present qualitative results obtained from interviews with robotic welding companies based in Norway
- Present a system description of the welding cell available at NTNU

1.2. Method

Referring to the objectives defined above, a large part of this report presents a "state of the art" review of the welding industry, which required extensive research to obtain qualitative data. The literature study was primarily based on the book "Welding Robots: Technology, System Issues and Application" by Norberto Piers, Altino Loureiro, and Gunnar Bolmsjö, which provides a detailed description of welding robots. As the book was published in 2006, further research has been performed to obtain the true picture of the industry today. The qualitative data found online have been validated through the following criteria:

- Robot manufacturer as author; many robot manufacturers explain important concepts within automated welding. However, they might be biased when explaining methods and/or technology.
- References and date published; even though the article was published relatively recently, the listed references might not have been.
- Author objectiveness; the author might prefer a certain method, technology, or robot manufacturer.

In order to obtain a true picture of the industry, as is today, interviews with different robotic welding companies were conducted. As with the literature study, qualitative data was the aim of the questionnaire. The information obtained was validated by narrowing the list of participants to companies based in Norway. This ensures somewhat equal conditions for the interviewees, which provides a safer ground for making assumptions based on the results obtained. Furthermore, only experienced welding robot programmers were interviewed. The questions should not be affected by how we, as the interviewers, perceive the industry. Therefore the questions were formulated to be short and concise while allowing for interpretation.

The questions asked are listed below. The first question was included in order to verify the answers to the subsequent questions. No minimum requirement was set for the number of years of experience, as all information was regarded as interesting.

The questions, as well as the interviews, were prepared and conducted in collaboration with Vebjørn Bergsholm Bjørhovde. The results obtained will, therefore, be equal in both our reports. The template sent to the interviewees is shown in Figure C.1 in Appendix C.

- How many years of experience do you have with welding robots?
- How do you program welding robots?
- Do you use sensors with your welding robots? If yes, which?
- Which welding techniques do you use with your robots?
- What is the biggest challenge you encounter while programming welding robots?
- To what degree is one able to fully automate the welding process? Does one often have to complete the weld manually afterward?

1.3. Outline of thesis

The report is structured as follows. Chapter 3 provides a "state of the art" review of the welding industry. Chapter 4 provides a system description of the welding cell at disposal for this project. The results obtained from the literature and the interviews conducted, and a discussion of these findings are given in Chapter 5 and Chapter 6 respectively. Chapter 7 concludes on the report and comments on future work.

Chapter 2.

Preliminaries

This chapter aims to provide the reader with a basic understanding of a robot manipulator; more specifically, "Robot Kinematics" is explained. The reader will be introduced to both forward- and inverse kinematics, as well as other important concepts such as velocity kinematics, singularities, and redundancy. With an understanding of all these concepts, one is able to both plan and control the movement of a robot manipulator. Industrial robots with six degrees of freedom will be the main focus in this chapter, as such robots are the topic of the successive chapters.

2.1. Rotation and translation matrices

Before introducing robot kinematics, two important matrices will be presented; the rotation matrix and the translation matrix. These matrices are essential when describing motions. To explain the concept of rotation matrices, consider first two frames; a space frame {s} and a body frame {b} which is rotated 90 degrees about the z_s axis, as shown in Figure 2.1. The orientation of the body frame {b} with respect to the space frame {s} can be described by the vectors x_b = (0,1,0), y_b = (-1,0,0) and z_b = (0,0,1). These vectors can be represented in a rotation matrix denoted R_{sb}, as shown in Equation (2.1). The subscript "sb" represent the reference frame, {s} in this case, and the frame to be transformed with respect to the reference frame, {b} in this case, respectively. The set of all 3×3 rotational matrices is called "The special orthogonal group $SO(3)$". The rotational matrices $R \in SO(3)$ are subjected to two conditions; (i) $R^{\mathrm{T}}R = I$ and (ii) det $R = 1$ [32].

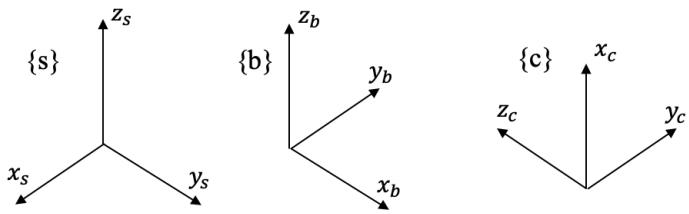


Figure 2.1.: Three frames {s}, {b}, and {c} with different orientations. Adapted from: [32]

$$R_{sb} = \begin{bmatrix} \mathrm{x}_b & \mathrm{y}_b & \mathrm{z}_b \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.1}$$

The rotation matrix in Equation (2.1) contains nine elements, but the space describing an orientation is three dimensional [32]. The elements of the rotation matrix are subjected to six constraints, summarised in condition (i) described above. This condition states that the matrix R must be orthogonal. An $n \times n$ matrix R is orthogonal if the relationship $R^{\mathrm{T}}R = I$ holds, R^{T} and I being the matrix R transposed and the identity matrix, respectively [42]. Condition (ii) contains the "special" case of the $SO(3)$ group. Whereas condition (i) ensures $\det R = \pm 1$, condition (ii) states that the determinant of all matrices R must be equal to $+1$. This implies the use of the right-hand rule to determine positive and negative rotation about an axis.

Rotation matrices are commonly used for three purposes [32], the first one being to represent an orientation. Referring to Figure 2.1 and Equation (2.1), the rotation matrix R_{sb} represents the orientation of the body frame {b} with respect to the space frame {s}. A second way to take advantage of the rotation matrix is to change reference frame. Imagine a third frame, {c}, with a different orientation than {s} and {b}, as shown in Figure 2.1. Suppose one wants to express the {b} frame in {c} coordinates, as opposed to {s} coordinates. The rotation matrix R_{cb} can then be found by the following matrix multiplication $R_{cb} = R_{cs}\, R_{sb}$, due to a cancellation principle: if the second subscript of the first matrix and the first subscript of the second matrix are equal, these cancel each other.

The third way to utilize the properties of the rotation matrix is to rotate a vector or a frame. Again, referring to Figure 2.1, by studying frame {b} with respect to frame {s} one can see that the former is obtained from the latter by a rotation

of 90 degrees about z_s. This can be expressed as $R_{sb} = R = \text{Rot}(z, 90°)$. This rotation matrix can be used in pre- and post-multiplications to rotate any frame. Say R_{sc} represents the orientation of frame {c} in {s} coordinates. If one pre-multiplies this matrix by the rotation operator $R = \text{Rot}(z, 90°)$, frame {c} would be rotated about the z axis corresponding to the first element in the subscript of R_{sc}, z_s in this case. The rotated frame can be represented as $R_{sc'} = R\ R_{sc}$, still expressed in {s} coordinates. For post-multiplication, the rotation about the z axis would correspond to the last element in the subscript of R_{sc}; z_c, and the rotated frame becomes $R_{sc''} = R_{sc}\ R$, also still expressed in {s} coordinates [32].

The angular velocity of frame {b} expressed in {s} can be represented by a unit vector, $\hat{\omega_s}$, and the speed of rotation about it, $\dot{\theta}$. The angular velocity is given as $\omega_s = \hat{\omega_s}\ \dot{\theta}$. Further, the linear velocities of the axes of frame {b} are all a function of the angular velocity and the respective axes:

$$\dot{x}_b = \omega_s \times x_b, \tag{2.2}$$
$$\dot{y}_b = \omega_s \times y_b, \tag{2.3}$$
$$\dot{z}_b = \omega_s \times z_b\,. \tag{2.4}$$

An important concept called the "skew-symmetric matrix" simplifies the mathematical computations involving cross-product of vectors. Say $x = [x_1 x_2 x_3]^{\mathrm{T}} \in \mathbb{R}^3$. One can define a 3×3 skew-symmetric matrix representation of x, denoted $[x]$, shown in Equation (2.5), which satisfies $[x] = -[x]^{\mathrm{T}}$. The set of all these 3×3 skew-symmetric matrices is called $so(3)$ [32].

$$[x] = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}. \tag{2.5}$$

The rotation matrix R usually refers to an orientation of the body frame relative to the space frame, so the subscript can be excluded. Therefore, by applying the notation of a skew-symmetric matrix explained above, a general expression for the relationship between $\dot{R}_{sb}$ and the angular velocity, ω_s, can be expressed as

$$\dot{R} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} \end{bmatrix} = [\omega_s]R. \tag{2.6}$$

From the equations above, one can deduce the following relations:

$$\omega_b = R^{-1}\omega_s = R^{\mathrm{T}}\omega_s \qquad [\omega_b] = R^{-1}\dot{R} = R^{\mathrm{T}}\dot{R}$$
$$\omega_s = R\omega_b \qquad [\omega_s] = \dot{R}R^{-1} = \dot{R}R^{\mathrm{T}}$$

In order to describe a motion consisting of both rotation and translation, yet another group is required; the special Euclidean group $SE(3)$. The special Euclidean group is also known as the group of homogeneous transformation matrices T in $\mathbb{R}^3$. The set of all these 4×4 transformation matrices T is called $SE(3)$. The transformation matrix T is shown in Equation (2.7) where $R \in SO(3)$ and the column vector p is in $\mathbb{R}^3$ [32].

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_1 \\ r_{21} & r_{22} & r_{23} & p_2 \\ r_{31} & r_{32} & r_{33} & p_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.7}$$

The R matrix in Equation (2.7) above is the rotation matrix presented earlier in this chapter. The vector p describes the position of the origin of the frame to be represented relative to the reference frame[32]. The last row of the transformation matrix is added for computational simplicity. Just as with the rotation matrix R, the transformation matrix T has three common applications, the first one being to represent a configuration. Referring to Figure 2.2, frame {b} is rotated by some rotation matrix R_{sb} relative to the space frame {s}, and translated by a vector p. The transformation matrix T representing frame {b} relative to frame {s} is found as

$$T_{sb} = \begin{bmatrix} R_{sb} & p_1 \\ 0 & 1 \end{bmatrix}. \tag{2.8}$$

The second application for the transformation matrix is to change reference frame. This is analogous to changing the reference frame with a rotation matrix. Referring to Figure 2.2, if we know the transformation matrices T_{sb}, calculated with R_{sb} and p_1, and T_{bc} calculated with R_{bc} and p_2, the transformation matrix representing frame {c} relative to frame {s} can be found as $T_{sc} = T_{sb}\, T_{bc}$.

The third way to utilize the transformation matrix is to displace a vector or a frame. Any given configuration can be achieved by first translating and then rotating a frame. Mathematically, this can be expressed as

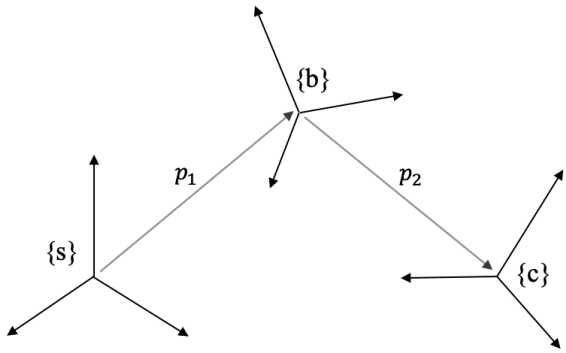


Figure 2.2.: Frames {s}, {b}, and {c} with different positions and orientations relative to each other. Adapted from: [32]

$$T = \text{Trans}(p)\,\text{Rot}(\hat{\omega},\theta) = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} & & & 0 \\ & & e^{[\hat{\omega}]\theta} & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.9}$$

Referring to frames {s} and {b} in Figure 2.2, the transformation matrix representing frame {b} relative to frame {s} is given as T_{sb}, and the displacement can be found by either pre- or post-multiplying the transformation matrix T_{sb} with T (2.9). Whether the variables p and $\hat{\omega}$ in Equation (2.9) are expressed in {s} or {b} depends on the order of the multiplication. When pre-multiplying, TT_{sb}, both p and $\hat{\omega}$ are expressed in {s}. This implies a rotation about the axis represented by $\hat{\omega}$, followed by a translation defined by p, both in the {s} frame. When post-multiplying, $T_{sb}T$, both p and $\hat{\omega}$ are expressed in {b}. This implies a translation defined by p followed by a rotation about the axis represented by $\hat{\omega}$, both in the {b} frame.

The 4×4 rotation matrix $\text{Rot}(\hat{\omega},\theta)$ in Equation (2.9) contains an element $e^{[\hat{\omega}]\theta} \in SO(3)$. This is the so-called "matrix exponential" which is defined as follows:

$$\text{Rot}(\hat{\omega},\theta) = e^{[\hat{\omega}]\theta} = I + \sin\theta[\hat{\omega}] + (1-\cos\theta)[\hat{\omega}]^2 \in SO(3). \tag{2.10}$$

An analogous representation for the transformation matrix in $SE(3)$ is given as

$$T = e^{[\mathcal{S}]\theta} = \begin{bmatrix} e^{[\omega]\theta} & (I\theta + (1 - \cos\theta[\omega]) + (1 - \sin\theta)[\omega]^2)v \\ 0 & 1 \end{bmatrix}, \tag{2.11}$$

for $||\omega|| = 1$, and for $\omega = 0$ and $||v|| = 1$ the expression becomes

$$T = e^{[\mathcal{S}]\theta} = \begin{bmatrix} I & v\theta \\ 0 & 1 \end{bmatrix}. \tag{2.12}$$

2.2. Robot kinematics

A robot manipulator consists of n joints connected by $n+1$ links, where actuators provide motive power allowing the links to move. The number of movable joints determines the robots degree of freedom, (DOF), where at least six DOF is required in order to fully describe an objects position in space [40]. Further, the number of joints determines the dimension of the configuration space, which contains all possible configurations of the robot. A robot with six DOF, implies a configuration space of dimension six. The task-, also called cartesian space of a robot, is where the task to be performed is expressed, and its dimension is determined by the number of variables required to describe the position of the end-effector. For a six DOF robot, the position and orientation of the end-effector is described by the coordinate system {x,y,z} and the rotation around each of the axes, often referred to as roll, pitch, and yaw. This implies a task space of dimension six. In fact, the dimension of the task space can not be higher than six, whereas the dimension of the configuration space can. However, if the dimension of the configuration space is greater than the dimension of the task space, the robot is said to be redundant. Redundancy is explained in Section 2.7. Whereas the task space expresses the tasks of a robot´s end-effector, the workspace specifies all the possible configurations of the end-effector [32].

Robot kinematics is a crucial tool in both understanding and controlling the motion of a robot. The kinematic model is used to describe the robot´s motion, excluding the forces required to achieve these motions [24]. Within kinematics, one differs between forwards- and inverse kinematics. While forward kinematics uses known information about the joint angles to calculate the position of the end-effector, inverse kinematics calculates the joint angles based on the desired position of the end-effector. A simple sketch is shown in Figure 2.3 to illustrate the concept. Forward kinematics problems are quite straight-forward to calculate, while inverse kinematics offers much more complex and time-consuming calculations.

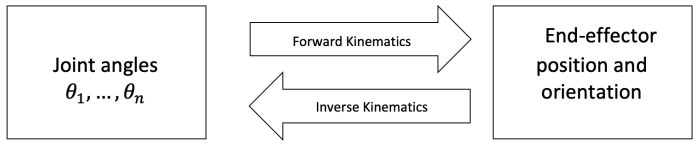


Figure 2.3.: Simple illustration comparing forward kinematics and inverse kinematics. Adapted from: [28]

2.3. Forward kinematics

Using known information about the joint angles of the robot to calculate the position and orientation of the end-effector is called forward kinematics. As the variables in the equations are known, forward kinematics always yields a solution, and the solution is unique. There are two main methods for calculating the forward kinematics for a given robot manipulator; Denavit-Hartenberg and Product of Exponentials [32].

2.3.1. Denavit-Hartenberg

There exists two versions of the Denavit-Hartenberg method, the one described in this section is the so-called "Modified Denavit-Hartenberg" [32]. This method describes each joint and link of the robot with four parameters; ϕ_i, d_i, a_{i-1}, and α_{i-1}, for i = 1 to n, n being the number of joints. The parameters describe the joint angle, link offset, link length, and link twist, respectively. The method then involves attaching a coordinate frame $\{\mathrm{x}_0, \mathrm{y}_0, \mathrm{z}_0\}$ to $\{\mathrm{x}_n, \mathrm{y}_n, \mathrm{z}_n\}$ to each joint, where the z_i axis points in the direction of the rotational/linear movement [5]. Each link transform is represented by a homogeneous transformation matrix T_i^{i-1} as shown in (2.13), where four elementary transformations are made; a rotation around z, a translation along z, a translation along x and a rotation around x [32].

$$\begin{aligned} \mathrm{T}_i^{i-1} &= \mathrm{Rot}(\hat{\mathrm{x}}, \alpha_{i-1})\,\mathrm{Trans}(\hat{\mathrm{x}}, a_{i-1})\,\mathrm{Trans}(\hat{\mathrm{z}}, d_i)\,\mathrm{Rot}(\hat{\mathrm{z}}, \phi_i) \\ &= \begin{bmatrix} \cos\phi_i & -\sin\phi_i & 0 & a_{i-1} \\ \sin\phi_i\ \cos\alpha_{i-1} & \cos\phi_i\ \cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i\ \sin\alpha_{i-1} \\ \sin\phi_i\ \sin\alpha_{i-1} & \cos\phi_i\ \sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i\ \cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{2.13}$$

As previously stated, only four parameters are required to express the forward kinematics when applying the Denavit-Hartenberg method. With the Product of Exponentials, six parameters are required in order to describe the displacement

in terms of orientation and position. The Denavit-Hartenberg method introduces two constraints on the placement of each coordinate frame $\{x_i, y_i, z_i\}$, which results in unique values for ϕ_i, d_i, a_{i-1}, and α_{i-1}. The first constraint states that axis x_i intersects axis z_{i-1} and the second that axis x_i is perpendicular to axis z_{i-1}. Each of the four parameters is then found by analyzing link i with respect to link $i-1$. Usually, the parameters are placed in a table, as shown in Table 2.1 where $i = 1$ to n, n being the number of joints. The solution to the forward kinematics problem is found by multiplying the n transformation matrices, as shown in Equation (2.14). This gives the position and orientation of the end-effector frame with respect to the base frame [28].

Table 2.1.: Example set-up of table with Denavit-Hartenberg parameters, where subscript x is some value between zero and n

i	α_{i-1}	a_{i-1}	d_i	ϕ_{i-1}
1	0	0	0	θ_1
⋮	⋮	⋮	⋮	⋮
i=n	$\pm$ 90°	L_{i-x}	0	θ_n

$$T^{\text{base}}_{\text{EndEffector}} = T^0_1 \dots T^{i-1}_i \tag{2.14}$$

2.3.2. Product of Exponentials

While the Denavit-Hartenberg method is quite straight-forward, it can be time-consuming. When calculating the forward kinematics using the Product of Exponentials, (PoE), one simply has to define two frames; a fixed base frame {s} and an end-effector frame {b}. It is, however, convenient to assign a frame to each of the robot´s n joints, where the z_n axis of each frame points in the direction of positive rotation [32]. The end-effector frame is described by a 4×4 matrix, M, which is determined when the robot is at its zero position, meaning all joint angles are equal to zero. The first three columns of M are determined by comparing $\{x_b, y_b, z_b\}$ to the base frame $\{x_s, y_s, z_s\}$. If for example x_b points in the negative direction of z_s, the first column of M would be $[0 \quad 0 \quad -1 \quad 0]^T$. The fourth and last column of M is set by evaluating the respective link lengths required to move the base frame $\{x_s, y_s, z_s\}$ to the end-effector frame $\{x_b, y_b, z_b\}$. The forward kinematics with the PoE formula is calculated as shown in Equation (2.15), where i = n, n being the number of joints.

$$T(\theta) = e^{[\mathcal{S}_1]\theta_1} \dots e^{[\mathcal{S}_{n-1}]\theta_{n-1}} \, e^{[\mathcal{S}_n]\theta_n} M. \tag{2.15}$$

As can be seen from Equation (2.15), the PoE formula requires one to define so-called "screw axes", $\mathcal{S}_i$. Each screw axis is a column vector with six elements, determined by ω_i and v_i, resulting in the expression: $\mathcal{S}_i = (\omega_i, v_i)$, for $i = 1$ to n, n being the number of joints [32]. The vector ω_i contains three elements and describes the i^{th} joint rotation with respect to the base frame. If the axis of rotation for join i points in the direction of -y_s, then ω_i = (0,-1,0). The vector v_i is found by the cross-product between -ω_i and q_i, where q_i is a vector with three elements describing the i^{th} joint translation with respect to the base frame. If for example link 2 has length L_1 in positive x_s direction, while the translation in y_s and z_s is zero, then q_2 = (L_1, 0, 0). The screw axes $\mathcal{S}_i$ in Equation (2.15) are expressed in matrix exponential form, which are found as shown in Equation (2.12). The solution to the forward kinematics problem for any given joint angle θ_i is given by Equation (2.15) with the respective screw matrices in exponential form $e^{[\mathcal{S}_i]}$ and the M matrix [32].

The above derived forward kinematics with the PoE is called the Space form, implying that the screw axes are represented in the base frame. Another representation of the PoE is the Body form, where the screw axes are represented in the end-effector frame; $\mathcal{B}_i = (\omega_i, v_i)$, for $i = 1$ to n, n being the number of joints [32]. With the body form, the M matrix representing the end-effector configuration at zero position is found by following the same procedure as with the space form above. The same applies for the respective screw axes expressed in the end-effector frame, but the vectors ω_i and v_i are determined with respect to the end-effector frame. If the axis of rotation of joint i points in the direction of y_b, then ω_i = (0,1,0). The vector v_i results from the cross-product of -ω_i and q_i, just as with the space form. The solution to the forward kinematics calculated in body form is given as shown in Equation (2.16), where the screw axes expressed in exponential matrix form, $e^{[\mathcal{B}_i]}$ are found as in Equation 2.12.

$$T(\theta) = Me^{[\mathcal{B}_1]\theta_1} \dots e^{[\mathcal{B}_{n-1}]\theta_{n-1}}\, e^{[\mathcal{B}_n]\theta_n}. \tag{2.16}$$

2.4. Inverse kinematics

Inverse kinematics transforms the position and orientation of the end-effector from the Cartesian-, or task space to the joint space, which is represented by the joint angles. Referring to Section 2.3.2, the solution to the forward kinematics problem using PoE was given as Equation (2.15). With inverse kinematics, one seeks to obtain the angles required to reach the desired end-effector position. The inverse kinematics problem can be expressed mathematically as

$$X = T(\theta) \tag{2.17}$$

where X represents the desired end-effector configuration [32]. As opposed to forward kinematics, which always yields one unique solution, inverse kinematics can give several valid joint angles for one and the same end-effector position. This means that there exist multiple configurations of the robot manipulator while the end-effector position remains the same. However, the desired solution to the inverse kinematics problem is the one that minimizes the joint motion while ensuring that the robot does not collide with itself [24]. As with forward kinematics, inverse kinematics problems can be solved using different methods. However, in contrast to forward kinematics, the choice of method does not depend on preference but rather the ability to achieve a solution. Further, regardless of the choice of method, one might not be able to find any solution to the inverse kinematics problem. The robot´s workspace was introduced at the beginning of this chapter as a space containing all reachable points for the robot manipulator. For inverse kinematics problems where X, the end-effector configuration, lies outside the robot´s workspace, there will not exists any solution [10].

2.4.1. Analytical inverse kinematics

Solving an inverse kinematics problem analytically is mathematically challenging, and the complexity increases with the number of joints. Ultimately the analytical inverse kinematics problem might become unsolvable. There is no "one method" when solving an analytical inverse kinematics problem; the mathematical computations depends on the robot manipulator in question. The general approach can be explained as follows: First, one needs to obtain the position of the end-effector by solving the forward kinematics problem. From the transformation matrix representing the pose of the end-effector, one extracts equations and solve these with respect to the joint angles. The equations obtained from the forward kinematics are nonlinear. Solving for the angles to obtain the inverse kinematics may therefore be difficult, with increasing complexity as the number of joints increase.

The number of joints influences the computational challenge, but also their geometric arrangement [34]. If the robot in question has a spherical wrist, the analytical inverse kinematics problem can be solved by decoupling the problem into inverse position and inverse orientation. A spherical wrist is a term describing a robot manipulator whose three wrist joints intersect at a single point. Most 6 DOF industrial robots have such spherical wrists [47]. Decoupling the inverse kinematics problem simplifies the computation [32]. The first three angles θ_1, θ_2, and θ_3 are obtained by the process explained above. With these angles derived, one can solve the inverse orientation by modifying Equation (2.15) into

$$e^{[\mathcal{S}_4]\theta_4}e^{[\mathcal{S}_5]\theta_5}e^{[\mathcal{S}_6]\theta_6} = e^{-[\mathcal{S}_3]\theta_3}e^{-[\mathcal{S}_2]\theta_2}e^{-[\mathcal{S}_1]\theta_1}XM^{-1} \tag{2.18}$$

Even though it is possible to obtain a solution to the analytical inverse kinematics problem for a six DOF industrial robot, the method is demanding and time-consuming. An alternative approach to solving the inverse kinematics for a given robot is a numerical approach.

2.4.2. Numerical inverse kinematics

Using numerical methods to solve the inverse kinematics problem is usually applied to robot manipulators where an analytical solution is unavailable. As stated in Section 2.3, one can always find the forward kinematics solution to any given robot manipulator. If one knows the desired end-effector configuration as well, one can simply adjust the angles so that the solution to the forward kinematics problem matches the desired end-effector configuration. This is usually achieved through an iterative process, such as the Newton-Raphson method [32]. This method is based on making an initial guess of the joint angles as well as determining two positive error allowance; one for the orientation and one for the linear position of the end-effector. The resulting angles must give an end-effector configuration satisfying the allowed errors. The initial guess of the joint angles must be sufficiently close to the solution for the iterations to converge, i.e., providing a solution.

2.5. Velocity kinematics

The previous sections of this chapter have been concerned with the position and orientation of the end-effector, expressed as forward- and inverse kinematics problems. However, the motion required to achieve these configurations involves translational and rotational movement. This is velocity kinematics.

The linear and angular velocity of an end-effector can be described by two components, ω and v, both column vectors with three elements. This results in a column vector with six elements, denoted $\mathcal{V}$. The column vector $\mathcal{V}$, as shown in Equation (2.19), describing the velocity of the end-effector is called the twist, or spatial velocity [32]. The twist $\mathcal{V}$ is determined by following the same procedure as with the screw axes in Section 2.3.2, the difference being that the screw axes in Section 2.3.2 were determined while the robot was at its zero position, implying

$\theta = 0$. For the twist, $\mathcal{V}$, the screw axes depend on θ.

$$\mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathbb{R}^6. \tag{2.19}$$

Further, the matrix representation of a twist can be written as

$$[\mathcal{V}] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} \in se(3). \tag{2.20}$$

Note that the above equations, (2.19) and (2.20), are written in general form. These expressions are valid for representing the twist in both space form and body form, called spatial twist and body twist, respectively. It can be shown that the relationship between the spatial twist and the body twist can be written as

$$[\mathcal{V}_b] = T^{-1}\dot{T} = T^{-1}[\mathcal{V}_s]T, \tag{2.21}$$

and

$$[\mathcal{V}_s] = T[\mathcal{V}_b]T^{-1}, \tag{2.22}$$

where the matrix T is the transformation matrix derived in Section 2.1. Writing out Equation (2.22) yields the following relationship between the spatial twist and the body twist

$$\begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = \begin{bmatrix} R & 0 \\ [p]R & R \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}. \tag{2.23}$$

The 2×2 matrix in Equation (2.23) pre-multiplying $\mathcal{V}_b$ is called the adjoint representation of T, denoted $[\mathrm{Ad}_T]$. This matrix is useful when changing between frames, where subscript T denotes the transformation matrix expressed in space form, T_{sb}, or body from, T_{bs}.

An important concept within robot kinematics must be addressed: the Jacobian matrix $J(\theta)$. The Jacobian is a matrix that provides a relationship between the joint velocities, $\dot{\theta}$, and the tip velocity vector, v_{tip} [32]. The velocity vector v_{tip} can be expressed in several ways; in the following section, the end-effector velocity will be represented by the twist $\mathcal{V}$. The relationship between the joint velocities and the twist is shown in Equation (2.24). For a robot manipulator the Jacobian

is given as $J(\theta) = [J_1(\theta)\ ..\ J_n(\theta)]$, where n is the number of joints. This implies that the number of columns in the Jacobian is equal to the number of joints in the robot manipulator. Further, $J_i(\theta)$ is the twist $\mathcal{V}_i$ when the corresponding $\dot{\theta}_i = 1$ and all other joint velocities are equal to zero.

$$\mathcal{V} = J(\theta)\dot{\theta} \tag{2.24}$$

As with the screw axis in Section 2.3.2, one can define a space Jacobian, $J_s(\theta)$, and a body Jacobian, $J_b(\theta)$. For the former, the elements of $J_s i(\theta)$ are set by the respective screw axis expressed in {s} frame. For the latter, the elements of $J_b i(\theta)$ are set by the respective screw axis expressed in {b} frame [32].

As mentioned above, the screw axes, $\mathcal{S}_i$ and $\mathcal{B}_i$, of a robot are determined while the robot is at its zero-position, implying all joint angles are equal to zero. However, the Jacobian $J(\theta)$ is defined for any arbitrary value of θ. The space Jacobian $J_s(\theta)$ for a robot manipulator with n joints is defined as

$$\mathcal{V}_s = J_s(\theta)\dot{\theta} = [J_{s1}\ J_{s2}(\theta)\ ...\ J_{sn}(\theta)]\dot{\theta}, \tag{2.25}$$

where $J_{s1} = \mathcal{S}_1$, and

$$J_{si}(\theta) = [\mathrm{Ad}_{e^{[\mathcal{S}_1]\theta_1 ... e^{[\mathcal{S}_{i-1}]\theta_{i-1}}}}]\mathcal{S}_i, i = 2, ..., n. \tag{2.26}$$

Here, the adjoint mapping $\mathrm{Ad_T}$ is used to represent the new screw axis with some arbitrary values of the joint angles, and the transformation matrix T is given as $\mathrm{e}^{[\mathcal{S}_1]\theta_1}...\mathrm{e}^{[\mathcal{S}_{i-1}]\theta_{i-1}}\mathcal{S}_i$. The same reasoning applies for the body Jacobian $J_b(\theta)$ which, for a robot manipulator with n joints, is defined as

$$\mathcal{V}_b = J_b(\theta)\dot{\theta} = [J_{b1}(\theta)\ ...\ J_{bn-1}(\theta)J_{bn}]\dot{\theta}, \tag{2.27}$$

where $J_{bn} = \mathcal{B}_n$, and

$$J_{bi}(\theta) = [\mathrm{Ad}_{e^{-[\mathcal{B}_n]\theta_n ... e^{-[\mathcal{B}_{i+1}]\theta_{i+1}}}}]\mathcal{B}_i, i = 1, ..., n-1. \tag{2.28}$$

2.6. Singularities

Singularities, when talking about robots, refers to a configuration of the robot manipulator which prohibits the end-effector from moving in certain directions [32]. This implies that when a robot manipulator is at a singularity, it loses one or

more of its degrees of freedom. Any robot manipulator with six degrees of freedom will have such singularities, at which its mobility will be limited. However, the complexity and type of singularity depend on the type of joints, the number of joints, and how the joints are configured [34].

Referring to Section 2.5, the Jacobian gives information about a robot manipulators singularities. For a six DOF robot, the resulting Jacobian matrix will be 6×6. Singularities for the six DOF robot will be where the Jacobian is not of maximum rank.

2.7. Kinematically redundant

A kinematically redundant robot manipulator is a manipulator that consists of more than six joints, i.e., $n > 6$. In the introduction to this chapter, it was stated that at least six DOF was required in order to fully describe the end-effector´s configuration in space. Further, the task space of any given robot manipulator can not have a dimension larger than six. For a robot with $6 + n$ DOF, the n successive joints are excess in terms of describing the end-effector configuration; they are redundant. However, the excess joints are useful in obstacle avoidance and for optimizing objective functions [32].

Chapter 3.

The automated welding industry

This chapter presents the automated welding industry from its origin till today. The intent is to provide a general understanding of the automated welding industry before introducing one of the main topics of research today, which is constraint-based robot programming. The reader will be introduced to the motivations behind the shift from manual to robotic welding, as well as the technological aspects making automated welding possible.

3.1. Motivations for automating a welding process

The use of industrial robots started in the mid-1950s and has evolved into a billion-dollar business [27, 43]. An industrial robot can be defined as "a programmable, mechanical device used in place of a person to perform dangerous or repetitive tasks with a high degree of accuracy." [5]. The first welding robots implemented in production lines were mainly used for large, long-run applications. Advances within the technology during the 1990s, such as collision detection, load identification, optimized programming languages, and more [25], made it possible to implement robots in smaller and more sophisticated welding operations. Today, industrial robots in welding operations is one of the most popular application of robotics in the world [45], meaning that automated welding operations very much do exist today. A clarification of the word "automated" in this context is needed. For a welding operation to be classified as automated, it is sufficient that part of the process is mechanized. If the welding process is fully automated, all parts of the process must be executed by a robot from start to finish [9].

Whereas manual welding can be traced back to around 3000 years BC [25], automated welding is a relatively new invention. The motivation for automating the

welding operation can be divided into economical- and security factors. Welding robots operate at a higher efficiency as compared to manual welders. According to the American Welding Society, the welding robot can increase arc efficiency up to 65%. While the actual speed of the welding robot might be the same as for the manual welder, the time in between each weld is not. The robot can perform these changes between welds at a higher speed than the manual welder and thereby improve the arc efficiency drastically [44].

Another important argument for automating the welding operation is the increase in repeatability and accuracy compared to manual welding. Repeatability, when talking about robotics, refers to the robot's ability to reach the same point over and over, whereas accuracy refers to the deviation between the desired point and the achieved point. The two concepts are illustrated in Figure 3.1. According to The Welding Institute, the repeatability of a typical industrial robot used in welding operations is $\leq$ 0.05 mm [30]. Further, a robot is able to follow its programmed path with an accuracy of 0.100 mm [7]. These qualities are essential in welding operations, and the industrial robot is superior to human welders.

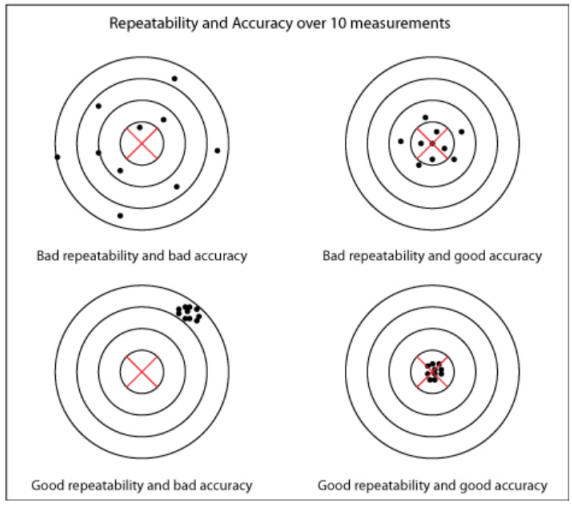


Figure 3.1.: Illustration of repeatability and accuracy plotted against each other. Source: [26]

Robots are also able to handle a larger workload as they, in theory, can work around the clock. These factors can contribute to a higher production rate due to increased efficiency and the elimination of human errors, which again can provide an increase in profit. For small to medium production volumes, automated welding productions have the best cost per unit performance [27]. This relation is shown in Figure 3.2. By replacing manual labor with automated labor, a decrease in

cost related to human operators can be achieved. There are, however, some factors that influence this. The time required to program the welding robot may be significantly higher than the time required to execute the manual weld [35]. Therefore, for small-scale operations, it might not be economically justifiable to implement robotic welding. Further, a different skill set is required to program and operate the robots, which would require the industry to educate their existing workforce or possibly replace them.

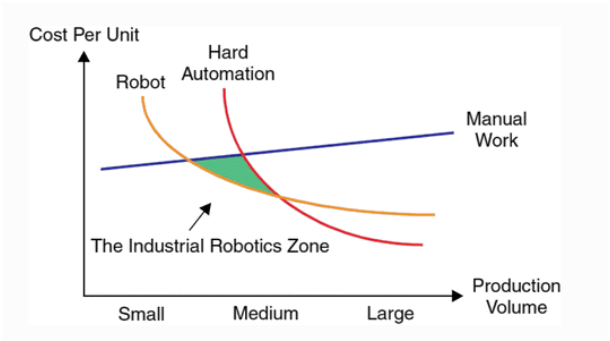


Figure 3.2.: Illustration of cost per unit versus production volume. Source: [27]

The above-mentioned security factors are related to the health risk welding operations pose on manual operators. The Labour Department of Hong Kong identified five hazards for manual welders; fire and explosion-, electrical- physical- respiratory- and other related hazards [12]. By automating the process, or part of the process, these risks are significantly reduced. However, the addition of a robot introduces a new safety hazard - a moving robot. This hazard is usually dealt with by enclosing the robot in some way, elaborated on in Section 3.2.

There are some issues related to automating the welding process [27]. Firstly, the company will face a higher initial cost due to the required equipment regarding the robot itself, the welding equipment, and any necessary security enclosure. Secondly, the loss of flexibility is a factor to consider as the robot is unable to adjust itself to any changes or unseen events in the same way a human welder can. The time required to program the robot can be demanding, so the production volume must justify the programming time. Further, certain segments in the welding industry introduce workspace constraints, such as pressure vessels, interior tanks, and ship bodies. The implementation of robotic welding in such areas might be problematic.

3.2. Welding robots

There are mainly two types of industrial robots applied in welding operations; rectilinear- and articulated robots [9]. Rectilinear robots have limited movement as they are restricted to linear movement along the three axis X, Y, and Z, where each subsequent prismatic joint is perpendicular to the former. The wrist can normally rotate, and the working zone forms a cube. Articulated robots resemble a human arm with rotational joints, making them more flexible. Articulated robots such as “Kuka”, “Fanuc”, “Panasonic”, “Motoman”, “Cloos”, and “Daihen” are commonly applied in welding operations. An illustration of a rectilinear robot is shown in Figure 3.3, and an illustration of an articulated robot is shown in Figure 3.4. Many different welding techniques exist which have their own set of requirements. Nevertheless, the basic concept of welding robots remains the same.

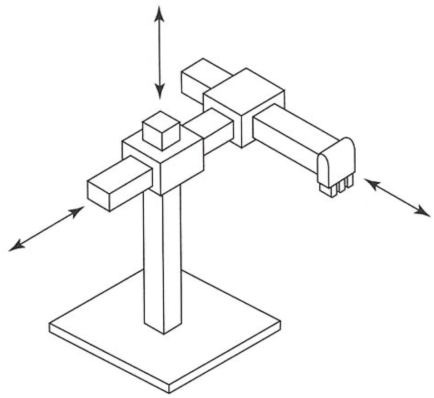


Figure 3.3.: Illustration of a rectilinear robot. Source: [14]

A robotic welding cell consists of two subsystems; the welding equipment and the actual robot [25]. The welding equipment implemented in automated welding differs from those applied in manual welding. As mentioned in Section 3.1, robots are in theory able to work around the clock, implying that the welding equipment must be designed as to withstand such high duty cycles. Further, the welding equipment must communicate with the robot through some kind of software [21]. The welding equipment consists of a welding torch attached to the wrist of the robot, a power source, and a wire feeder. The torch is attached to the robot while the power source and wire feeding system are stationed next to the robot. The required wires which deliver both power and wire to the torch can be mounted on or implemented within the robot arm, allowing for a more flexible robot. The welding power source supplies electricity, which is converted to heat, utilized to melt the metal. Several systems exist for the wire feeding, such as dispensing arms, mechanical turntables, and motorized dispensers [9]. The robot has its own

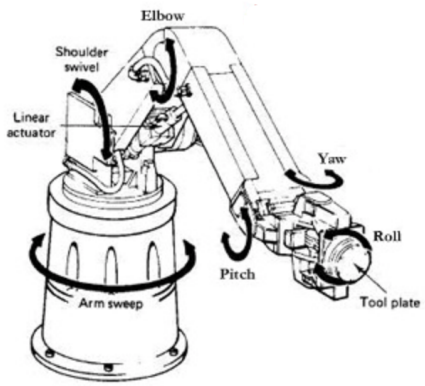


Figure 3.4.: Illustration of an articulated robot. Source: [38]

power supply and controller. The two subsystems can communicate through the control system.

As well as the robot manipulator and the welding equipment, a positioner is usually installed in the cell. A rotating positioner allows for more complex welding, where the torch and rotating positioner cooperate during weld [9]. A fume hood or similar ventilation set-up must be installed to reduce the toxic fumes and maintain good air quality. Finally, an installation separating the operator from the welding operation must be present. Many different solutions exist, such as curtains, screens, and enclosures. These can be either colored or see-through, protecting from both damaging light and spatter. Whereas curtains only serve as protection against dangerous light and spatter, an enclosure with a lock mechanism at the door can be utilized to protect the operator from the moving robot manipulator as well as the hazards that accompany welding.

A general welding process can be divided into three phases: the preparation phase, the welding phase, and the analysis phase [25]. The preparation phase contains all necessary preparations prior to the actual welding. In this phase, the operator sets all parameters, places the piece to be welded, and prepares the apparatus, i.e., the robot, the power supply, and the robot program.

Robots are mainly programmed in two ways; online or offline [36]. Whereas online programming is executed near the welding cell in direct contact with the welding robot, offline programming is done through an external computer that does not need to be near the welding cell. The main difference between the two is that online programming causes production downtime seeing as the robot has to be taken out of production while being programmed. With offline programming, the robot does not have to be taken out of production as the programming is

done through an external device. Industrial robot languages are elaborated on in Section 3.5.

The welding phase is where the welding operation is executed. While performing the welding operation, the robot should be able to maintain a torch orientation that follows the desired trajectory [25]. Further, the robot should be able to, amongst others, perform seam-tracking and change welding parameters in real-time. The ability to do so depends on the implementation of sensors, elaborated on in Section 3.4. The analysis phase involves examining the weld and determining whether quality demands are met or if changes to the two previous phases are required. If advanced sensors such as 3D laser cameras are implemented, this task can be executed online during the previous phase [27].

3.3. Welding techniques

As mentioned above, several different welding techniques are applied in the industry of welding robots today. Robotics Online has developed a list of the most common robotic welding techniques; arc welding, resistance welding, spot welding, TIG welding, MIG welding, laser welding, and plasma welding [46].

Arc welding is a method that utilizes the heat produced by the electric arc, which arises between the workpiece and electrode, to melt metal. This process requires high repeatability, with no more than $\pm$ 0.5 mm arc position deviation between each consecutive workpiece [50]. Further, the filler wire is often slightly bent, which increases the risk of the deviation becoming unacceptably large. However, arc welding robots account for 40% of all industrial robots, and the segment is growing rapidly [9].

TIG welding, also known as Gas Tungsten Arc Welding (GTAW), is a form of arc welding that uses a tungsten electrode and a stream of inert gas or a mixture of gasses to protect the weld pool from air contamination [25]. Figure 3.5 illustrates the concept of GTAW. MIG, which is a subtype of Gas Metal Arc Welding (GMAW), is similar to GTAW. The major difference between the two is that GTAW uses a non-consumable electrode, tungsten, and a "filler metal" is necessary to produce the weld pool. With GMAW, a consumable electrode is used; hence no extra filler metal is needed. Figure 3.6 illustrates the concept of GMAW. Whereas GMAW is considered the easier method, GTAW is preferred when, for example, the weld seam will be visible because of its clean appearance [16].

Resistance welding utilizes a current which is passed between two pieces of metal in pressure, resulting in a pool of metal. The current is turned off while the pressure is maintained, resulting in a fusion of the two pieces. Spot welding

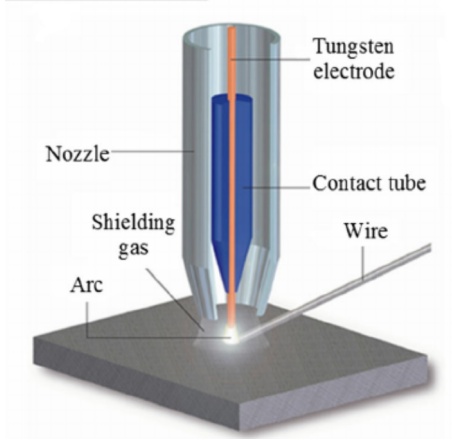


Figure 3.5.: Illustration of gas tungsten arc welding. Source: [13]

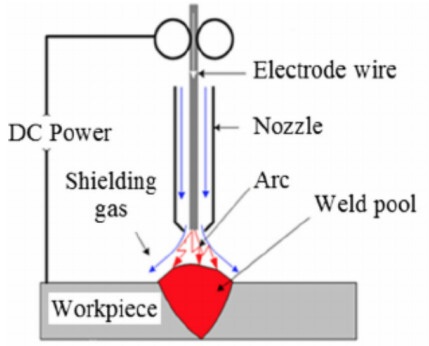


Figure 3.6.: Illustration of gas metal arc welding. Source: [13]

is a form of resistance welding where pressure is applied by two shaped copper electrodes through which current is supplied. The heat that occurs causes the two metal pieces to melt. The current is turned off, and the two pieces are joint together while still under pressure [31]. The technique is illustrated in Figure 3.7. Spot welding has been implemented in automated welding operations since the beginning due to a relatively large position error allowance for the sheets to be joined. Whereas the repeatability of today's welding robots is ± 0.05 mm, it was closer to 1 mm for the earlier generations [50].

Laser welding utilizes a laser beam, which provides a highly concentrated heat source where the beam is pointed [49]. The area under the laser beam absorbs the light, and the electrons in the exposed area become excited, and melting of the metal will result. The advantages of using this method are that the resulting weld

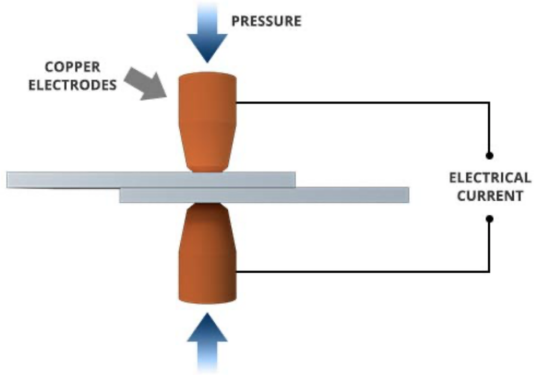


Figure 3.7.: Illustration of resistance spot welding. Source: [31]

is precise and clean. Only a very concentrated part of the workpiece is exposed to heat, no electrode is necessary, and the process is easily automated. Due to its preciseness, laser welding is often utilized in the automotive-, jewelry- and medical industries [15].

Robotic Plasma Arc Welding (PAW) is similar to TIG, which uses a non-consumable tungsten electrode. However, the arc that arises between the electrode and the workpiece is constricted by a plasma nozzle, which also directs a flow of plasma gas. The shielding gas is provided on the outside of the nozzle [48]. Figure 3.8 illustrates the concept of PAW.

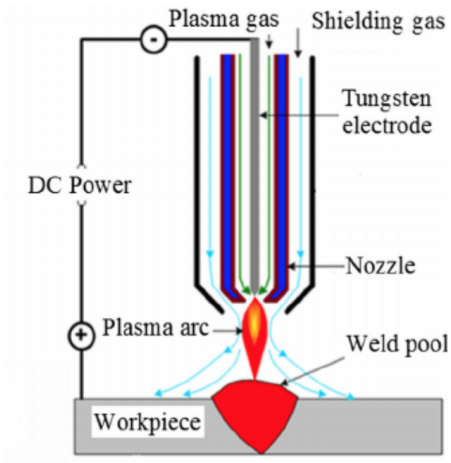


Figure 3.8.: Illustration of plasma welding. Source: [13]

Others worth mentioning are Friction Stir Welding (FSW) and Cold Metal Transfer (CMT). Friction stir welding differs from the ones already mentioned because the welding process takes place in the solid phase of the metals. This method does not need filler metals or shielding gasses [25]. CMT is a welding method developed from GMAW, but the method produces less heat than GMAW by retracting the wire once a short circuit is detected [17]. This results in heat being introduced only in short periods during arc-burning. This is highly desirable because heat can cause unwanted changes in the properties of the metal to be welded [51].

3.4. Sensor technology

The need for sensor technology arises with any automated system, and the welding cell is no exception. A manual welder can, for example, examine the workpiece for discrepancies prior to weld and adjust the welding process in real-time. The implementation of sensor technology allows for a welding robot to more closely resemble a manual welder. With the implementation of sensors in robotic welding operations, one introduces the possibility of detecting and measuring process features, and parameters [27]. One can divide the sensors used in automated welding operations into two categories; sensors for technological parameters and sensors for geometrical parameters.

Sensors for technological parameters refer to sensors that measure parameters related to the welding, such as arc voltage, welding current, and wire feed speed. Sensors for geometrical parameters include optical sensors and through-arc sensing, to name a few of the most used [25]. While the technological sensors are implemented for monitoring and/or controlling the welding process, geometrical sensors are used in order to examine the workpiece to be welded [27]. The technological sensors give information about the welding process, but the geometrical sensors enable the path of the robot to be adjusted if there exist deviations from the nominal path.

Consider a welding robot whose path is pre-programmed, and no sensors are implemented. Due to the robot's repeatability explained in Section 3.1, the robot will weld exactly where it is programmed to, with small deviations. However, the piece to be welded might not be placed at the exact same position every time. According to The Lincoln Electric Company, a welding robot expects the piece to be welded to be placed at the same position each time, with a position error of $\pm$ 0.127 mm [9]. In addition to the possible position error above, deficiencies in the material, as well as property changes due to heat application, also occur [29]. This can result in a weld seam with a position deviation. Sensors are implemented in order to account for such deviations through three main functionalities; seam-finding, seam-tracking, and/or part scanning [29].

The most common sensors implemented in robotic welding are; touch, through-arc, laser, and vision [29]. Through-wire touch sense is a so-called "contact sensor" as it obtains information by being in physical contact with the surface of the workpiece. This sensor does not require any access equipment mounted on the robot as the welding wire is used to detect the conductive surface and make electric contact. A low-voltage circuit is used to detect the weld joint as the robot moves along the workpiece. An illustration of through-wire touch sensing is shown in Figure 3.9. The obtained information from the sensor is used to adjust the path of the robot prior to weld [8]. The disadvantages of this method are the required wire feeder and cutter, as well as being time-consuming as the robot has to examine the entire workpiece prior to the actual weld. Further, this sensor method is best suited when the piece to be welded has defined edges.

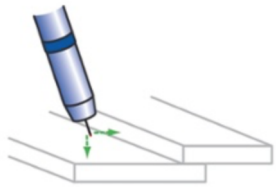


Figure 3.9.: Through-wire touch sense. Source: [29]

As opposed to the through-wire sensor, which is a contact sensor, the through-arc seam tracking technology is a non-contact sensor. A sensor placed near the welding power source measures the arc characteristics while the robot moves back and forth across the weld joint [8]. The current increases as the distance between the arc and the workpiece decreases, and the current decrease as the distance increase. This implies that the desired position of the robot along the weld joint is where the measured current is at its minimum. An illustration of through-arc seam tracking is shown in Figure 3.10. This method is less time consuming than the through-wire sensor, as the measured arc characteristics are obtained in real-time, i.e., while the robot is executing the weld. Drawbacks with this method include the restrictions that accompany the wavy motion of the torch; the weld joint dimension must be over some value. Further, it is not possible to determine start- and end position with the sensor method as the arc must be on in order to obtain sensor information [27].

Laser sensors are another example of non-contact sensors. A laser and sensor are mounted on the robot, which together provide both position and orientation of the workpiece. A line is projected onto the workpiece, and the information gathered is based on the distortion of the line [8]. An illustration of a laser sensor is shown in Figure 3.11. Even though this technique is based on obtaining information

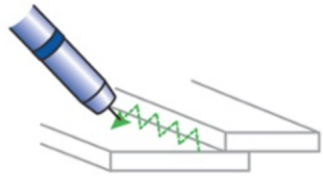


Figure 3.10.: Through-arc seam tracking. Source: [29]

about the workpiece and adjusting the path prior to the weld, it ensures faster cycle times than, for example, the through-wire touch sensing as the robot does not have to move along the entire workpiece. Laser sensors are applicable for workpieces with both thickness and joint gap less than 1 mm. Problems might arise with reflective surfaces. However, by implementing software that allows for noise-canceling, one can manage to overcome such problems. Further, laser sensors are more expensive than the previous two, and some limitations regarding the movement of the robot manipulator may arise due to the placement of the sensor.

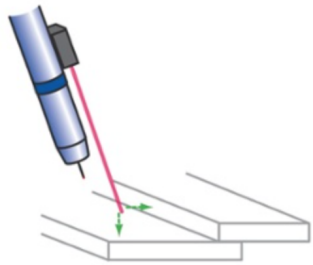


Figure 3.11.: Laser sensor. Source: [29]

Vision sensors include both 2D and 3D cameras. A 2D camera is able to obtain both the position and orientation of the workpiece, however, only in two dimensions. This may introduce problems if pieces to be welded are placed on top of each other, for example. An illustration of a 2D camera is shown in Figure 3.12. A 3D camera exhibits the same qualities as a 2D camera, with the inclusion of a laser which provides the three-dimensional image [8]. The complexity increases with the implementation of 3D cameras due to the need for software such as image recognition, which further requires personnel employing knowledge of such software. Therefore, 3D cameras are not normally used in stand-alone welding operations, but rather in larger production lines of which the welding robot is a part [29]. 2D cameras are normally sufficient, but the implementation of camera

sensors in general faces some challenges. Cameras are sensitive to both light conditions and the surface of the workpiece. Further, the harsh environment caused by the welding operation is problematic for cameras.

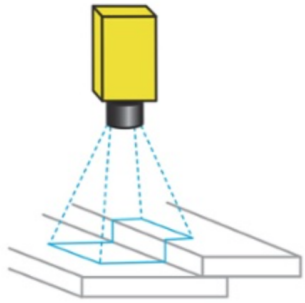


Figure 3.12.: 2D camera. Source: [29]

3.5. Industrial robot languages

As explained briefly in Section 3.2, several methods exist for programming a robot. The two main categories are; online- and offline programming. Online programming of a robot includes walk-through and lead-through, where walk-through requires the operator to manually move the torch to its desired positions, which are then recorded [27]. Lead-through utilizes a teach-pendant, which serves as an interface between the robot and the operator. This method involves a human operator who moves the robot through so-called "jogging" with control keys or a joystick to its desired position and then records this position. This procedure is repeated for all desired positions.

The teach-pendant is usually equipped with different modes, each with its own set of constraints [52]. In teach mode, the operator prepares and teaches the robot the job to be done. Referring to Section 3.2, the preparation phase is normally executed with the teach-pendant in teach mode. Further, play mode enables the operator to playback the programmed job. Both teach- and play mode have velocity constraints as to restrict the robot from moving at high velocities, ensuring safety while constructing and adjusting the program. The last mode is called remote mode. Again, referring to Section 3.2, this is the mode in which the welding phase is executed. While in remote mode, the program can be executed at full speed. To ensure safety while in remote mode, some sort of safety barrier is usually installed in order to separate the robot and the operator, as explained in Section 3.2. An example of such an installation is explained in Section 4.4.

An industrial robot can generally move in three different motions; jointed, linear, and circular [9, 33]. The jointed movement is programmed with the command "MoveJ" which instructs the robot to move from A to B on all axis simultaneously. This movement is suitable when the actual path from A to B is less important. If the path from A to B is important, the linear motion programmed as "MoveL" is preferred. This instructs the robot to move from A to B while maintaining torch angles. Whereas the linear motion is preferred when the actual path is important, the jointed motion is safer because the robot cannot collide with itself while executing the path. The circular motion can be programmed as "MoveC" where one defines the start- middle- and endpoint, and a circular path is generated. A small example is presented to illustrate the concept of programming a robot using the teach pendant. The desired path to be programmed is shown in Figure 3.13, and the associated commands for achieving this path are shown in Table 3.1.

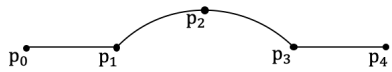


Figure 3.13.: Example of simple path with half circle movement. Adapted from: [52]

Table 3.1.: Instructions required to achieve path as shown in Figure 3.13. Adapted from: [52]

Point	Instruction
p_0	MoveJ/MoveL
p_1	MoveC
p_2	
p_3	
p_4	MoveJ/MoveL

Another way to program robots is through offline programming (OLP). OLP allows for the robot's trajectory to be programmed using an external computer instead of the teach-pendant, used in online programming. As mentioned in Section 3.2, this method reduces downtime as the programming is done outside of production, as opposed to online programming where the robot has to be taken out of production while being programmed [39]. OLP can be divided into two main categories; Text-based programming and Graphical offline programming [37]. Text-based programming involves writing a program on a standard keyboard in some text editor and uploading the program to the robot controller once finished. Even though preferred programming language is subjective, the most common program-

ming languages used in robot programming are C++ and Python [36]. Further, many robot manufacturers have developed their own software that is compatible with their robots. Whereas ABB has its robot language called RAPID, Kuka has KRL, Fanuc uses Karel, and Universal Robots uses URscipt.

In other words, industrial robot languages are not universal. The existence of all these different robot languages can be challenging as one has to learn a new language when switching between one robot brand and another. This potential exposure to different robots is related to the fact that robot manufacturers often specialize their robot for certain areas, resulting in the need for different robots within the same production [22]. Different offline programming software has been developed independently of robot brand, which are compatible with several robots. An example of such a software is RoboDK. RoboDK provides an extensive software that is compatible with more than five hundred robot arms from forty different manufacturers, including robot manufacturers who provide their own software, such as ABB and Kuka [23]. The software allows for one to upload 3D models of the tool as well as the workpiece. The desired trajectory can be edited both graphically, i.e., moving the robot in the scene editor and in a text-based program with a preferred programming language such as Python and C++, to name a few.

3.6. Constraint-based programming

As explained in Section 3.5, there are currently two methods for programming welding robots; online- and offline programming. These methods are sufficient in order to program a simple task such as welding of a straight line, where extensive software resources are available from both robot manufacturers as well as independent developers. A topic of research within the robotic industry today are the possibilities of introducing constraint-based programming in order to achieve more complex tasks [11]. Constraint-based programming involves representing a problem in terms of constraints and decision variables and then using a constraint solver to solve them [41]. The defined constraints might conflict and must, therefore, be assigned a certain priority [3]. This is due to the fact the conflicting constraints can not be satisfied at the same time, so the priority of the constraints determines which constraint to be satisfied in such a situation. Whereas the above-mentioned programming methods are specialized for repetitive tasks, constraint-based programming targets the need that arises from changing environment and fluctuating demand, namely a robot system that can be utilized in several different environments, without necessarily knowing all future applications [20].

An example of such a language is Expressiongraph-based Task Specification Language (eTaSL), based on the scripting language Lua [3]. This is a task specification

language, implying that the defined constraints are with regards to the task at hand. The language provides tools for describing how a given robot system has to interact with the sensors present, where this description of the interaction with sensors is based on constraint-based methodology. The controller, eTC, is divided into three layers; specification, solver, and numerical solver. The specification layer is where the constraints are defined by using the task specification language, eTaSL. The solver translates these constraints into a numerical optimization problem where, finally, the numerical solver solves the defined optimization problem.

This method of defining constraints and decision variables can be utilized to, for example, avoid collision between two robot manipulator arms with overlapping trajectories. Whereas traditional online- and offline programming aims to define collision-free trajectories, constraint-based programming can be utilized to avoid collision even though the trajectories collide. This quality can be exploited in robot systems with changing environments and unseen future applications.

Chapter 4.

System description of welding cell at NTNU

As mentioned in Section 3.2, there exist many different welding robots in the industry today. The welding robot at disposal for this project is the Yaskawa Motoman GP25-12. This chapter presents a system description of the welding cell as installed at NTNU, as shown in Figure 4.1. This includes the robot manipulator, the welding equipment, and the enclosure. Simple calculations on the robots forward- and inverse kinematics are presented in order to illustrate the concept explained in Section 2.

4.1. Industrial robot controller

It is important to distinguish between the controller and the teach pendant. The teach pendant provides easy access to the robot through a touchscreen display where one can both see and edit the available commands, obtain information about different joint variables, develop and edit programs, etc. The teach pendant, referred to as the programming pendant by Yaskawa [1], is equipped with a keyboard which allows for easy maneuvering in the form of, for example, "jogging", explained in section 3.2, and online programming. Further, a button is installed at the back of the pendant, which serves as a "live-man" switch by either being pressed- or released fully. This ensures safety while both testing and running programs. An additional emergency button is implemented in the upper left corner. Figure 4.2 shows the Motoman YRC1000 Industrial Robot Controller with the provided programming pendant in front of the controller.

The robot controller can be said to be the brain of the robot seeing as the actual control lies within the controller. Code, also referred to as programs, developed either online through the teach pendant or offline via an external computer, are

Figure 4.1.: Yaskawa Motoman GP25-12 with welding equiptment as installed at NTNU

exported to the controller via a communication port such as, for example, an Ethernet connection. The exported program is translated to physical motions [6] in the controller. This information is then sent to the robots Central Processing Unit, CPU, which enables the robot to both process and run the program [2].

4.2. The robot manipulator

The Yaskawa Motoman GP25-12 is a robot manipulator with six rotational joints connected by seven links, as can be seen in Figure 4.1. Referring to Chapter 2, this robot has six DOF. This implies that the GP25-12 has a configuration space and a task space of dimension six. As the dimension of the configuration space is equal to the dimension of the task space, this robot is not kinematically redundant. However, the robot will have configurations in which singularities arise [34]. Yaskawa Motoman labels each axis as S, L, U, R, B, and T, which stands for Swing or Swivel, Lower arm, Upper arm, Rotate, Bend, and Twist, respectively [4]. Six axes allow for the robot manipulator to exhibit the movement of a human arm. Each axis has limitations with regard to maximum motion range and maximum speed. The repeatability, discussed in Section 3.1, of this robot is

Figure 4.2.: Controller and programming pendant

± 0.03 mm. Further, the maximum work range is 2010 mm. The above-explained specifications as well as other important variables are summarized in Table A.2 and Table A.3 in Appendix A.2.

4.2.1. Kinematic calculations

As mentioned in Section 3.5, Python is a programming language commonly applied within robotics. The programming language Python has been utilized in this section, together with the code editing program Visual Studio Code (VS-code). One can import different Python modules in VS-code, allowing for programming customized to specific needs. When developing a program for robots in Python, an important library is "numpy", which provides important tools for, for example, array computations. Further, a library called "modern_robotics" provides many functions for calculating a robot's kinematics. A simple program calculating the forward- and inverse kinematics for the Yaskawa Motoman GP25-12 has been developed in order to illustrate the concepts explained in Section 2.

The zero-position for the GP25-12 is as shown in Figure A.1 in Appendix A.2, where the respective lengths from joint one to joint six can be identified as; l_1 = 505 mm, l_2 = 760 mm, l_3 = 200 mm, l_4 = 150 mm, l_5 = 275 mm, l_6 = 807 mm, and l_7 = 100. Note that this includes an offset between joint one and two, resulting in seven identified lengths. A simplified sketch of the robot has been

developed in order to illustrate the concept, shown in 4.3.

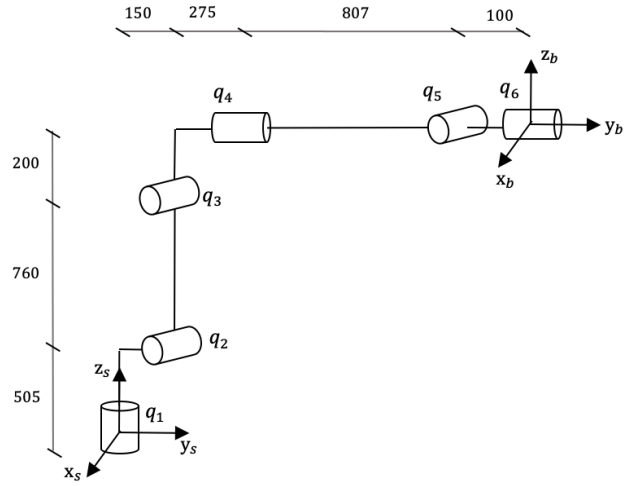


Figure 4.3.: Simplified sketch of Yaskawa GP25-12 in its zero-position

The global coordinate system is defined using the right-hand rule with positive x axis pointing out of the paper, positive z axis pointing upwards, resulting in positive y axis pointing in the direction of the spherical wrist. The M matrix representing the end-effector configuration when the robot is at its zero-position, explained in Section 2.3.2, is found to be as shown in Equation (4.1) with lengths l_1 - l_6 as above.

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & l_5 + l_5 + l_6 + l_7 \\ 0 & 0 & 1 & l_1 + l_2 + l_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1332 \\ 0 & 0 & 1 & 1465 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

However, with the welding gun mounted on the wrist, the new end-effector position will be at the tip of the welding gun. To obtain this configuration, the homogeneous transformation matrix described in Section 2.1 was used. Let {b} denote the frame at the wrist, and {t} denote the frame at the tip of the welding gun. By studying Figure 4.4 one can see that frame {t} relative to frame {b} is rotated some amount about the x_b axis and translated some amount in y- and z direction. These values were obtained from the programming pendant. The transformation matrix expressing the configuration of the new end-effector was

found to be

$$M_{\text{tool}} = T_{\text{bt}} = \begin{bmatrix} R_{\text{bt}} & p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.59 & 0.81 & 450.27 \\ 0 & -0.81 & 0.59 & 84.40 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.2}$$

where $R_{\text{bt}} = \text{Rot}(\text{x}, \theta) = \text{Rot}(\text{x}, -0.94)$ and $p = [\text{x}, \text{y}, \text{z}] = [0, 450.27, 84.40]$. Note that the angle given in radians is negative, due to the previously defined coordinate frame $\{\text{x}, \text{y}, \text{z}\}$ explained earlier in this section.

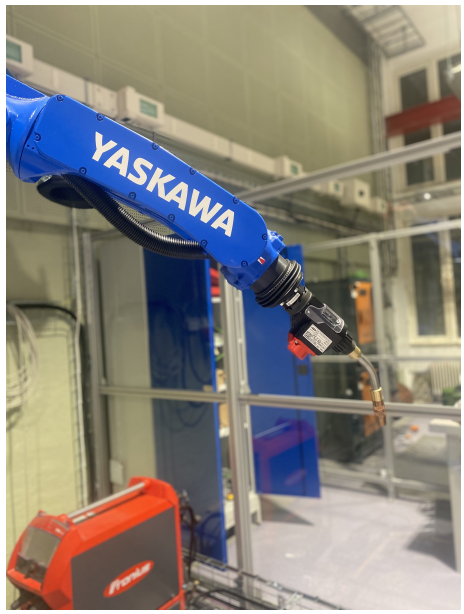


Figure 4.4.: Close-up of wrist and the attached welding gun

The screw axes represented in space form, $\mathcal{S}_i = (\omega_i, v_i)$, and in body form, $\mathcal{B}_i = (\omega_i, v_i)$, can be found as explained in Section 2.3.2. The vectors ω_i and v_i are presented in Table 4.1 in space form and Table 4.2 in body form. Each screw axis is the 6×1 column vector formed by the respective ω_i and v_i. The screw axes were calculated in VS-code as shown in Figure B.2 and Figure B.3 in Appendix B.

To compute the forward kinematics solution, two functions in the "modern_robotics" library can be utilized; FKinSpace and FKinBody. Both functions take as input the matrix M as well as a list of joint angles, named "thetalist" by default. The

Table 4.1.: Vectors ω_i and v_i for i=1 to 6 for Yaskawa GP25-12 represented in space form

i	ω_i	v_i
1	(0,0,1)	(0,0,0)
2	(1,0,0)	(0, l_1, -l_4)
3	(1,0,0)	(0, l_1+l_2, -l_4)
4	(0,1,0)	(-(l_1+l_2+l_3),0,0)
5	(1,0,0)	(0, l_1+l_2+l_3, -(l_4+l_5+l_6))
6	(0,1,0)	(-(l_1+l_2+l_3), 0, 0)

Table 4.2.: Vectors ω_i and v_i for i=1 to 6 for Yaskawa GP25-12 represented in body form

i	ω_i	v_i
1	(0,0,1)	(-(l_7+l_6+l_5+l_4),0,0)
2	(1,0,0)	(0, -(l_3+l_2), l_7+l_6+l_5)
3	(1,0,0)	(0, -l_3, l_7+l_6+l_5)
4	(0,1,0)	(0,0,0)
5	(1,0,0)	(0, 0, l_7)
6	(0,1,0)	(0, 0, 0)

list of angles will be a 1×6 vector for the GP25-12 as it has six joints. The last input depends on whether the screw axes were calculated in space form or body form. The function FKinSpace takes a list of screw axes calculated in space form, named "Slist" by default. The function FKinBody takes a list of screw axes calculated in body form, named "Blist" by default. These lists can be calculated in VS-code as shown in Figure B.2 and Figure B.3 in Appendix B. The resulting Slist and Blist for the GP25-12 are shown in Equation (4.3) and (4.4), respectively.

$$\text{Slist} = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1465 & 0 & -1465 \\ 0 & 505 & 1265 & 0 & 1465 & 0 \\ 0 & -150 & -150 & 0 & -1232 & 0 \end{bmatrix} \tag{4.3}$$

$$\text{Blist} = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ -1332 & 0 & 0 & 0 & 0 & 0 \\ 0 & -960 & -200 & 0 & 0 & 0 \\ 0 & 1182 & 1182 & 0 & 100 & 0 \end{bmatrix} \tag{4.4}$$

For inverse kinematics calculations, the library "modern_robotics" uses the Newton-Raphson method explained in Section 2.4.2. As with the forward kinematics, there exist two functions for computing the inverse kinematics; IKinSpace and IKinBody. Both functions take in the matrix M, a matrix describing the desired end-effector configuration, a list of guessed joint angles, and an error allowance "eomg" for orientation and "ev" for position. Again, the last input depends on whether the screw axes were calculated in space form or body form. IKinSpace takes Slist as input, and IKinBody takes Blist as input, both lists being equal to the ones derived above.

To calculate the inverse kinematics for the GP25-12, the transformation matrix given as the solution to the forward kinematics was set as the desired end-effector configuration, T = T_goal. Then, an initial guess of the joint angles was stored in a list called "thetalist_guess". The error allowance for orientation was set to 0.001, and the error allowance for position was set to 0.0001.

The two functions will output a list of angles required to obtain the desired end-effector configurations and a result of the iterations, which will be either "true" or "false". The result would be true if the iterations were able to find a solution within the defined error allowances and false if not. If the initial guess of angles is too far from the solution, the iterations will not converge, and the result will be false.

4.3. Welding equipment

A Fronius CMT welding system is installed with the Yaskawa robot manipulator. CMT, explained briefly in section 3.3, is a welding technique that utilizes very low heat input and as a result shields the material to be welded from property changes, amongst others. The CMT welding system from Fronius works by detecting a short circuit. During welding execution, the short current is supervised by the digital process control, and at occurrence, the welding wire is pulled back. This short circuit control results in low temperatures [17].

Figure 4.5.: Fronius TPS 400i

More specifically, the welding equipment from Fronius installed at the lab is the MIG/MAG power source TPS 400i as can be seen in Figure 4.5. The wire feeder is shown in Figure 4.6. The TSP 400i is equipped with a touch screen, which provides the operator with system information, and the operator is also able to perform adjustments such as welding parameters. The technical data for the TPS 400i is provided in Table A.1 in Appendix A.2.

The welding gun is attached to the robot´s wrist joint as an additional piece of equipment, where the tip of the welding gun makes out the end-effector of the robot manipulator. The wiring from both the wire feeder and the power source is gathered on the robot prior to the spherical wrist. The and the wires provided to the welding gun are implemented within the spherical wrist, allowing for a more flexible movement.

4.4. Welding cell enclosure

As explained in Section 3.2, different solutions for enclosing the welding cell exist. The Yaskawa motoman GP25-12 with Fronius TPS 400i is enclosed by see-through walls, as shown in Figure 4.7. A lock mechanism is installed at the door, which is implemented in the software of the robot. This functions as a safety measure where the operator won't be able to execute programs in remote mode unless the door is closed. This protects the operator from both dangers related to the welding operation, such as spatter and damaging light, and also the robot manipulator itself.

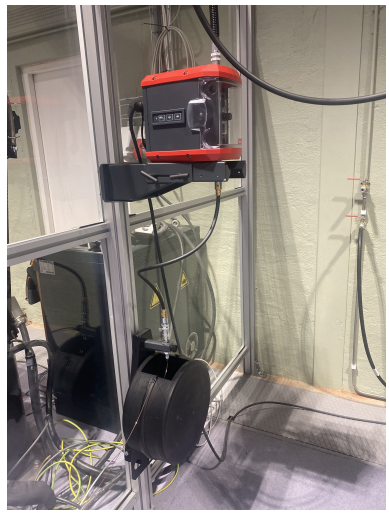


Figure 4.6.: Wire feeder

Figure 4.7.: Welding cell enclosure

Chapter 5.

Results

This chapter presents the findings obtained from both the "state of the art" review of the welding industry and the interviews held with experienced personnel in the industry.

5.1. Results from the literature

The results obtained from the literature are summarized in the list below.

- Robotic welding can increase arc efficiency by up to 65% as compared to manual welders, as well as provide a higher production rate and a reduction in errors related to manual welding.
- The health risk posed on human operators is lowered when implementing welding robots.
- The time required to program the robot must be economically justifiable. Programming a robot can be time-consuming and often exceeds the time required to perform the manual weld. Small-scale operations might not be able to economically justify automated welding.
- The initial cost when implementing robotic welding can be high and must be economically beneficial in the long-run.
- Implementation of welding robots can be problematic in areas with workspace constraints.
- Sensors are an essential part of a robotic welding system to make it comparable with manual welders.
- The most commonly applied sensors are; touch, through-arc, laser, and vision. Touch sensors are more time-consuming than through-arc as the

sensor information is obtained prior to welding. Through-arc sensors adjust the path in real-time but can not be used for start- and endpoint detection as the arc must be on to obtain sensor information. Both laser- and vision sensors are sensitive to reflective surfaces and present a higher cost than the touch- and through-arc sensors.

- Two main methods for programming welding robots today are online- and offline programming. The main drawback with online programming, as opposed to offline programming, is the break in production.
- Many robot manufactures provide software with their robots, specifically developed for their robots. This can be problematic for robot programmers who have to learn a new robot language when switching to a new robot brand.
- Constraint-based programming is a topic of research today and represents a new way of programming robots.

5.2. Results from interviews with personnel in the industry

The qualitative results obtained from the interviews with personnel in the robotic welding industry in Norway are presented in the same order as asked in the interviews. Three different interviews were obtained, where the answers from each interviewee are summarized below. The three interviewees are referred to as interviewee 1, interviewee 2, and interviewee 3.

How many years of experience do you have with welding robots?

The interviewees had an average of 4 years of experience.

How do you program welding robots?

All interviewees used online programming with teach-pendant as their main method, explained by this being the easiest way to program robots.

Interviewee 3 works mainly with programming of the teach-pendant, as to prepare this for the robotic welding programmer. The interviewee explained how UR-script and Polyscope, which is the graphical interface of UR, was used. Their target group is low-volume production with larger variance, which requires a manual programming process. One depends on having both the workpiece and robot at hand while programming. It was mentioned that the possibility of using OLP with CAD models was present but not implemented to this date.

Interviewee 1 and 2 used OLP as well as the teach-pendant. The ability to simulate the process prior to executing the weld was highlighted with this method. Further, it was stated that when using OLP, the software provided with the robot in question was utilized. Interviewee 1 explained how each OLP software provided by the robot manufacturer has its own unique design, making it difficult to switch between different robot brands.

Interviewee 2 explained how advanced parts requiring OPL were programmed by the customer themselves, as they wished to learn the process in order to be able to adjust the program for the next advanced part, different from the former. Interviewee 1 explained how OLP was sometimes used to program the robot in complex operations, followed by a touch-up using the teach-pendant. It was further pointed out that the time required to program a robot was closely related to the complexity of the task. Easy operations were quick to program, whereas more complex operations could take anywhere from ten minutes to two days.

Do you use sensors with your welding robots? If yes, which?

All interviewees said that they used sensor technology with their welding robots. Seam tracking sensor technology was highlighted by interviewee 1 and 2. Wheres interviewee 1 explained how they utilized voltage-sensing, where constant voltage was used to maintain a constant distance to the workpiece, interviewee 2 stated that through-arc sensing was mostly used with their robots. The voltage-sensing technology was mostly used with TIG, and not suitable with MIG.

Interviewee 3 explained how 3D cameras were used, combined with tough-sensing, in order to obtain the geometry of the piece to be welded. This technology was, however, only used prior to welding.

Which welding techniques do you use with your robots?

When asked about welding techniques, the interviewees all agreed that MIG/MAG welding was the most common method applied in the industry of welding robots, and they all used it with their robots. This was explained by its easy implementation and use, and based on customer request. Interviewee 1 explained how new solutions within the power supply have made it possible to further evolve MIG/MAG into techniques such as MIG pulse and CMT. It also allows for use on aluminum and titanium. Interviewee 2 expressed how MIG/MAG now could be used to weld aluminum down to 0.6 mm.

Further, all interviewees agreed that TIG was less used in the industry due to being more difficult to implement with welding robots. It is, therefore, usually implemented when the operation in question requires this method explicitly. Interviewee 3 expressed that they wanted to try TIG welding with orbital weld.

Interviewee 1 stated that spot welding is the most common method applied in

the industry today, whereas laser welding is increasingly used, especially in the car industry. Interviewee 2 stated that laser- spot- and friction stir welding is less used in Norway, more common in the automotive industry.

Interviewee 2 and 3 used Fronius´s welding equipment, where interviewee 3 also used Kempi. Interviewee 1 expressed how Fronius was one of the best brands on the market, providing automatic welding equipment.

What is the biggest challenge you encounter while programming welding robots?

With this question, there was a larger deviation between the answers obtained. Interviewee 1 listed the following problems related to the programming of a robot:

- Problematic when the complexity accompanying the task exceeds the programming knowledge of the operator
- When one has several large programs, it can be cumbersome to get them to communicate
- Input/output problems
- Sensors that does not work
- An older robot usually offer more problems due to having been "tampered" with for many years

Interviewee 2 explained how the pieces to be welded must be identical in order to obtain the correct weld, as the robot is unable to make adjustments the way a manual welder can. However, the interviewee stated that through-arc sensors are able to account for variations with the workpieces. Further, it was said that programming a welding robot is not more or less difficult than programming any other robot, but one has to take into account the welding parameters.

Interviewee 3 explained how their challenge lies within making a user-friendly interface that is intuitive. The interviewee stated that programming of a robot using the teach-pendant should be both quick and easy. Further, it was explained how welding parameters are a challenge when programming the teach-pendant, where it is often hired welding engineers by either them or their customers.

To what degree is one able to fully automate the welding process? Does one often have to complete the weld manually afterward?

Interviewee 1 explained how the degree of automation is closely related to the design of the welding cell. One is able to design a welding cell as to achieve a fully automated process. However, this can lead to problems with reaching certain areas, which again can result in some parts needing manual work afterward. Interviewee 2 explained that if one has to manually finish the weld due to inefficient

results, the customer is usually unsatisfied with the result,so the entire weld has to be redone by the robot.

Both interviewee 2 and 3 mentioned that one sometimes has to grind parts of the workpiece manually after a weld. Interviewee 2 explained that for welding processes that leave “slag”, which requires the operator to manually grind the workpiece afterward, one tends to execute the entire process manually, as this is more efficient. Further, interviewee 3 explained how they aim for the welding process to be 80/20, 80% automated and, 20% manual. It was explained how they provided small-scale welding robots that do not need enclosures and are easy to move around where needed, making them suitable for smaller companies.

Chapter 6.

Discussion

The industry of welding robots has grown substantially since the invention of the first industrial robot. Through the literature, it has become evident that a major challenge when implementing robots in welding operations is that no subsequent operation is exactly the same. This is explained by, for example, discrepancies in the workpiece as well as position errors. A welding robot without any sensors is, in theory, only able to act based on programmed commands. By implementing sensors, the ability to react arises. Vision sensors are widely implemented with robots in general but become problematic with welding robots due to the environment that follows any welding operation. It seems as though vision sensors are the latest technology within robotic sensors. However, if they can't be utilized during welding to collect information in real-time, the welding robot essentially fails to compete with manual welders. A combination of sensors might be the solution; vision sensors can be utilized prior to welding in order to obtain detailed information regarding position and orientation, while through-arc sensors can be utilized during welding to provide real-time information of the welding process.

Interviewee 3 said that they utilized 3D cameras and touch-sensing; however, only prior to welding. Both interviewee 1 and 2 only utilized non-contact sensors. None of the interviewees utilized a combination of sensors to obtain information both prior to and during welding. This might be explained by physical constraints. The implementation of sensors implies extra equipment mounted on the robot manipulator, which might reduce its mobility. Sensor technology has undoubtedly come a long way; however, it seems reasonable to assume that cameras capable of withstanding the harsh environment accompanying welding operations could have been achieved, based on the advancement in vision technology in general. One might ask whether the lack of such cameras is due to a lack of incentive to innovate in the supply chain, such as vision sensor manufacturers, or rather a lack of demand from the robotic welding industry itself. Even though the automated welding industry has evolved greatly since the invention of the industrial robot,

the advancement has been slow the recent years. This could be interpreted as the industry being satisfied with the technology currently available to the industry, seeing as automated welding has been successfully implemented in productions for decades.

Further, the literature seems to agree that the main drawback with using online programming as opposed to offline programming is the break in production while programming. However, from the answers obtained from the interviews, it became clear that their main preference was online programming with the teach-pendant. This finding encourages a discussion; why is online programming with the teach-pendant the preferred choice when the benefit from offline programming seems so apparent? Offline programming allows for continuous welding operations, which contribute to a higher production rate. Further, this method allows for simulations prior to welding, a highly desirable advantage, according to interviewees 1 and 2. However, interviewee 1 explained how offline programming is often followed by online programming with the teach-pendant. This leads to the assumption that the only experienced benefit from offline programming is the possibility to simulate. The teach-pendant is equipped with modes allowing for slow execution, which in essence can be compared to a simulation when the aim of the simulation is to ensure a collision-free trajectory. Further, the problem related to different robot languages was stated as being a challenge with offline programming when changing between different robot brands. It seems as if the theoretical benefit with offline programming might not be accurate in practice.

Interviewee 1 and 2 agreed that offline programming was introduced first when the problem at hand was complex. Interviewee 2 explained that their customers used offline programming for advanced parts in order to learn the process for the next advanced part, which differed from the former. Interviewee 3 explained how they programmed the teach-pendant as their target customer is low-volume production with large variance, which requires a manual programming process. From this, one might conclude that the common factor is the complexity that is introduced with a changing environment. These findings motivate the conclusion that the way to program a robot today might not be ideal, as the operator utilizes the less preferred method when the complexity of the task increase.

In order to make the automated welding industry even more versatile, a change seems necessary. Constraint-based programming was introduced as one of the topics of research within the robotic industry today. To the best of my knowledge, this has not been implemented successfully in a robotic welding operation. However, it has been developed for, for example, pick-and-place operations. Interviewee 2 explained how the programming of a welding robot is no different from programming any other robot, except for the welding parameters one has to account for. This can be interpreted as that constraint-based programming

should be applicable to welding robots as well.

While comparing the results obtained from the interviews with the literature, one important factor must be addressed; the extent of the conducted interview. The basic concept of robotic welding can be said to be universal; however, their application can not. The application of robots in welding operations is highly related to the industry in that particular country. This leads to the assumption that the discrepancies between practice and theory discussed above might be different if one, for example, conducted interviews with robotic welding personnel in Japan, where robotic welding is an essential part of the automotive industry.

Chapter 7.

Conclusion and further work

The shift from manual to automated welding can be said to be a success. Robots in welding operations are an essential part of many highly advanced production lines. However, there are some short-comings in the industry. The two main findings in this report are the challenges related to sensor technology as well as the programming of the robot when the task is no longer repetitive but rather new and ever-changing.

The results obtained from the interviews with experienced personnel in the industry were highly valuable as they highlighted some discrepancies between theory and practice. It is concluded that the extent of the questionnaire is too narrow to fully understand where the short-comings of the global industry lie today. A more extensive questionnaire focusing on robotic welding in different segments of the industry today should be conducted.

Further, it is concluded that there does not seem to be a strong motivation from the industry to innovate; while new methods such as constraint-based programming offer the possibility of a more versatile robot system, the interviewees seemed satisfied with the currently available technology. This means that there is a lack of incentive for the supply industry to invest in innovation. However, I believe that we will continue to see improvement within the robotic welding industry, based on the simple fact that robotic welding offers indispensable benefits. In addition, there are still health risks related to human operators, which will always serve as an incentive to develop new methods.

Improvements can be achieved by further developing sensor technology as to withstand the harsh environment that follows welding operations. While cost related to advanced technology is a factor to consider, a general trend is that prices decrease as demand and supply increase. It is concluded that the demand is present, which provides an incentive to the supply industry, which again could result in more affordable technology. Further, future research seems to lie within improving

the ways of programming robots to allow for easier completion of complex tasks as well as to make the robot system more versatile with regards to fluctuating demand and unseen applications.

References

[1] URL: https://www.yaskawa-global.com/product/robotics/about.

[2] URL: https://robotsdoneright.com/Articles/main-components-of-an-industrial-robot.html.

[3] Erwin Aertbeliën and Joris De Schutter. "Etasl/eTC: A Constraint-Based Task Specification Language and Robot Controller Using Expression Graphs". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2014, pp. 1540–1546. ISBN: 9781479969340.

[4] Yaskawa America. *Robotics Glossary*. URL: https://www.motoman.com/en-us/about/company/robotics-glossary.

[5] Robotic Industries Association. *Defining The Industrial Robot Industry and All It Entails*. URL: https://www.robotics.org/robotics/industrial-robot-industry-and-all-it-entails.

[6] Mathieu Bélanger-Barrette. *What is Included in Robotic Welding Systems?* Feb. 2016. URL: https://blog.robotiq.com/bid/72927/What-is-Included-in-Robotic-Welding-Systems.

[7] Tom Bonine. *Robotic Welding Advantages*. July 2015. URL: https://www.automation.com/en-us/articles/2015-2/robotic-welding-advantages.

[8] The Lincoln Electric Company. *Robotics: Joint Sensing Technologies*. URL: https://www.lincolnelectric.com/en-gb/support/process-and-theory/Pages/intelligent-robotic-detail.aspx.

[9] The Lincoln Electric Company. *THE DEFINITIVE GUIDE TO AUTOMATION For Welding and Metal Fabrication*. Feb. 2015. URL: https://www.lincolnelectric.com.cn/en/CustomApi/download?fileItemId=%7B75372F24-8DB6-45BA-83B9-DD04CBFE8732%7D&type=preview.

[10] Nikolaus Correll. *Advanced Robotics 4: Inverse kinematics*. Feb. 2012. URL: http://correll.cs.colorado.edu/?p=1958.

[11] Tinne De Laet, Johan Rutgeerts, Wilm Decré, Ruben Smits, Erwin Aertbeliën, Kasper Claes, and Herman Bruyninckx. "Constraint-based Task Specification and Estimation for Sensor-Based Robot Systems in the Presence of Geometric Uncertainty". In: *I. J. Robotic Res.* 26 (2007), pp. 433–455. DOI: 10.1177/0278364907078091O7.

[12] Labour Department. *Code of Practice - Safety and Health at Work for Manual Electric Arc Welding*. July 2002. URL: https://www.labour.gov.hk/text_alternative/pdf/eng/welding3.pdf.

[13] Donghong Ding, Zengxi Pan, Dominic Cuiuri, and Huijun Li. "Wire-feed additive manufacturing of metal components: technologies, developments and future interests". In: *International Journal of Advanced Manufacturing Technology* 81 (May 2015). DOI: 10.1007/s00170-015-7077-3.

[14] Industrial Electronics. *Rectilinear Robots*. Sept. 2007. URL: http://www.industrial-electronics.com/industrial-electricity-com/5_Rectilinear_Robots.html.

[15] Ben Ellis. *The 7 Types of Robotic Welding Processes*. Mar. 2019. URL: https://www.cyberweld.co.uk/the-7-types-of-robotic-welding-processes.

[16] Kevin Fairchild. *GMAW, GTAW, AND SMAW WELDING: WHAT YOU NEED TO KNOW*. Oct. 2019. URL: https://www.engineeredmechanicalsystems.com/gmaw-gtaw-and-smaw-welding-what-you-need-to-know/.

[17] Fronius International GmbH. *CMT – COLD METAL TRANSFER: THE COLD WELDING PROCESS FOR PREMIUM QUALITY*. URL: https://www.fronius.com/en/welding-technology/world-of-welding/fronius-welding-processes/cmt.

[18] Fronius International GmbH. *TPS/i*. URL: https://www.fronius.com/en/welding-technology/products/manual-welding/migmag/tpsi/tpsi/tps-400i.

[19] Yaskawa Europe GmbH. *Yaskawa*. URL: https://www.yaskawa.eu.com/products/robots/handling-mounting/productdetail/product/gp25_699.

[20] L. Halt, F. Nagele, P. Tenbrock, and A. Pott. "Intuitive Constraint-Based Robot Programming for Robotic Assembly Tasks* The research leading to these results has received funding from the European Unions Seventh Framework Programme FP7/2013-2017 under grant agreement n 608604 (LIAA: Lean Intelligent Assembly Automation) and Horizon 2020 Research and Innovation Programme under grant agreement n 688642 (RAMPup)." In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 520–526. DOI: 10.1109/ICRA.2018.8462882.

[21] T S Hong, M Ghobakhloo, and W Khaksar. “Comprehensive Materials Processing”. In: Oxford: Elsevier, 2014. Chap. 6.04 - Robotic Welding Technology, pp. 77–99. DOI: https://doi.org/10.1016/B978-0-08-096532-1.00604-X.

[22] RoboDK Inc. *Off-line Programming*. URL: https://robodk.com/blog/off-line-programming/.

[23] RoboDK Inc. *Simulate and Program your Robot in 5 easy steps*. URL: https://robodk.com/simulation.

[24] Jamshed Iqbal, Muhammad Ul Islam, and Hamza Khan. “Modeling and analysis of a 6 DOF robotic arm manipulator”. In: *Canadian Journal on Electrical and Electronics Engineering* 3 (2012), pp. 300–306. DOI: https://www.researchgate.net/publication/280643085_Modeling_and_analysis_of_a_6_DOF_robotic_arm_manipulator.

[25] Gunnar Bolmsjö J. Norberto Piers Altino Loureiro. *Welding Robots: Technology, System Issues and Application*. Springer, 2006.

[26] Ahmed Joubair. *What are Accuracy and Repeatability in Industrial Robots?* May 2016. URL: https://blog.robotiq.com/bid/72766/What-are-Accuracy-and-Repeatability-in-Industrial-Robots.

[27] Paul Kah, M Shrestha, E Hiltunen, and J Martikainen. “Robotic arc welding sensors and programming in industrial applications”. In: *International Journal of Mechanical and Materials Engineering* 10 (2015). DOI: 10.1186/s40712-015-0042-y.

[28] Serdar Kucuk and Z Bingul. “Industrial Robotics: Theory, Modelling and Control”. In: vol. December. Germany/ARS, Austria: Pro Literatur Verlag, 200§. Chap. 4, pp. 117–147.

[29] Josh Leath. *Knowing When Welding Sensors Make Sense*. Oct. 2018. URL: https://www.motoman.com/en-us/about/blog/knowing-when-welding-sensors-make-sense.

[30] TWI Ltd. *Robotic Arc Welding*. URL: https://www.twi-global.com/technical-knowledge/job-knowledge/robotic-arc-welding-135.

[31] TWI Ltd. *What is Spot Welding?* URL: https://www.twi-global.com/technical-knowledge/faqs/what-is-spot-welding.

[32] Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning and Control*. Cambridge University press, 2019.

[33] Kevin McWhirter. *Welding Robot Programmability*. Jan. 2012. URL: https://www.canadianmetalworking.com/canadianfabricatingandwelding/article/welding/welding-robot--programmability.

[34] Mecademic. *What are singularities in a six-axis robot arm?* URL: https://www.mecademic.com/resources/Singularities/Robot-singularities.

[35] Alex Owen-Hill. *The Simple Way to Flawless Robot Welding*. Aug. 2018. URL: https://robodk.com/blog/flawless-robot-welding/.

[36] Alex Owen-Hill. *What is the Best Programming Language for Robotics?* Mar. 2020. URL: https://blog.robotiq.com/what-is-the-best-programming-language-for-robotics.

[37] Alex Owen-Hill. *What Is the Best Way to Program a Robot?* Oct. 2018. URL: https://robodk.com/blog/program-robot-tips/.

[38] Kagan Pittman. *Articulated Robot Market Worth USD$79.58 Billion by 2022*. May 2016. URL: https://www.engineering.com/AdvancedManufacturing/ArticleID/12192/Articulated-Robot-Market-Worth-USD7958-Billion-by-2022.aspx.

[39] Delfoi Robotics. *Offline programming. Robot simulation and offline programming*. URL: https://www.delfoi.com/delfoi-robotics/robotics/.

[40] Motion Controls Robotics. *Unraveling Degrees of Freedom and Robot Axis: What does it mean to have a multiple axis pick and place or multiple axis robot?* URL: https://motioncontrolsrobotics.com/unraveling-degrees-of-freedom-and-robot-axis-what-does-it-mean-to-have-a-multiple-axis-pick-and-place-or-multiple-axis-robot/.

[41] Francesca Rossi, Peter van Beek, and Toby Walsh. "Chapter 1 Introduction". In: *Foundations of Artificial Intelligence* 2 (C 2006), pp. 3–12. DOI: 10.1016/S1574-6526(06)80005-2.

[42] Todd Rowland. *Orthogonal Matrix*. URL: https://mathworld.wolfram.com/OrthogonalMatrix.html.

[43] Mr. Shelly Singh. *Robotic Welding Market worth 5.96 billion USD by 2023*. July 2018. URL: https://www.marketsandmarkets.com/PressReleases/robotic-welding.asp.

[44] American Welding Society. *The robotic welding industry today*. Apr. 2019. URL: https://insights.globalspec.com/article/11712/the-robotic-welding-industry-today.

[45] Weria Khaksar Tang Sai Hong Morteza Ghobakhloo. "Robotic Welding Technology". In: (2014), pp. 3–34. DOI: https://doi.org/10.1016/B978-0-08-096532-1.00604-X.

[46] Robotics Online Marketing Team. *7 Common Types of Robotic Welding Processes and When They're Used*. Nov. 2017. URL: https://www.robotics.org/blog-article.cfm/7-Common-Types-of-Robotic-Welding-Processes-and-When-They-re-Used/72.

[47] Queensland University of Technology. *Inverse Kinematics and Robot Motion*. URL: https://robotacademy.net.au.

[48] TWI. *WHAT IS PLASMA WELDING?* URL: https://www.twi-global.com/technical-knowledge/faqs/faq-what-is-plasma-welding.

[49] Kashyap Vyas. *Laser Welding: Types, Advantages, and Applications*. Aug. 2019. URL: https://interestingengineering.com/laser-welding-types-advantages-and-applications.

[50] Klas Weman. "Welding Processes Handbook (Second Edition)". In: Woodhead Publishing, 2012. Chap. 15 - Mechanisation and robot welding, pp. 157–166. DOI: https://doi.org/10.1533/9780857095183.157.

[51] Ben Wiley. *What is Cold Metal Transfer and Is It More Efficient than MIG*. May 2017. URL: https://www.wileymetal.com/what-is-cold-metal-transfer-and-is-it-more-efficient-than-mig/.

[52] Inc Yaskawa America. *DX200 OPERATOR'S MANUAL FOR SPOT WELDING USING MOTOR GUN*. 2015. URL: http://assets.efc.gwu.edu/yaskawa-motoman/165297-1CD.pdf.

Appendix A.

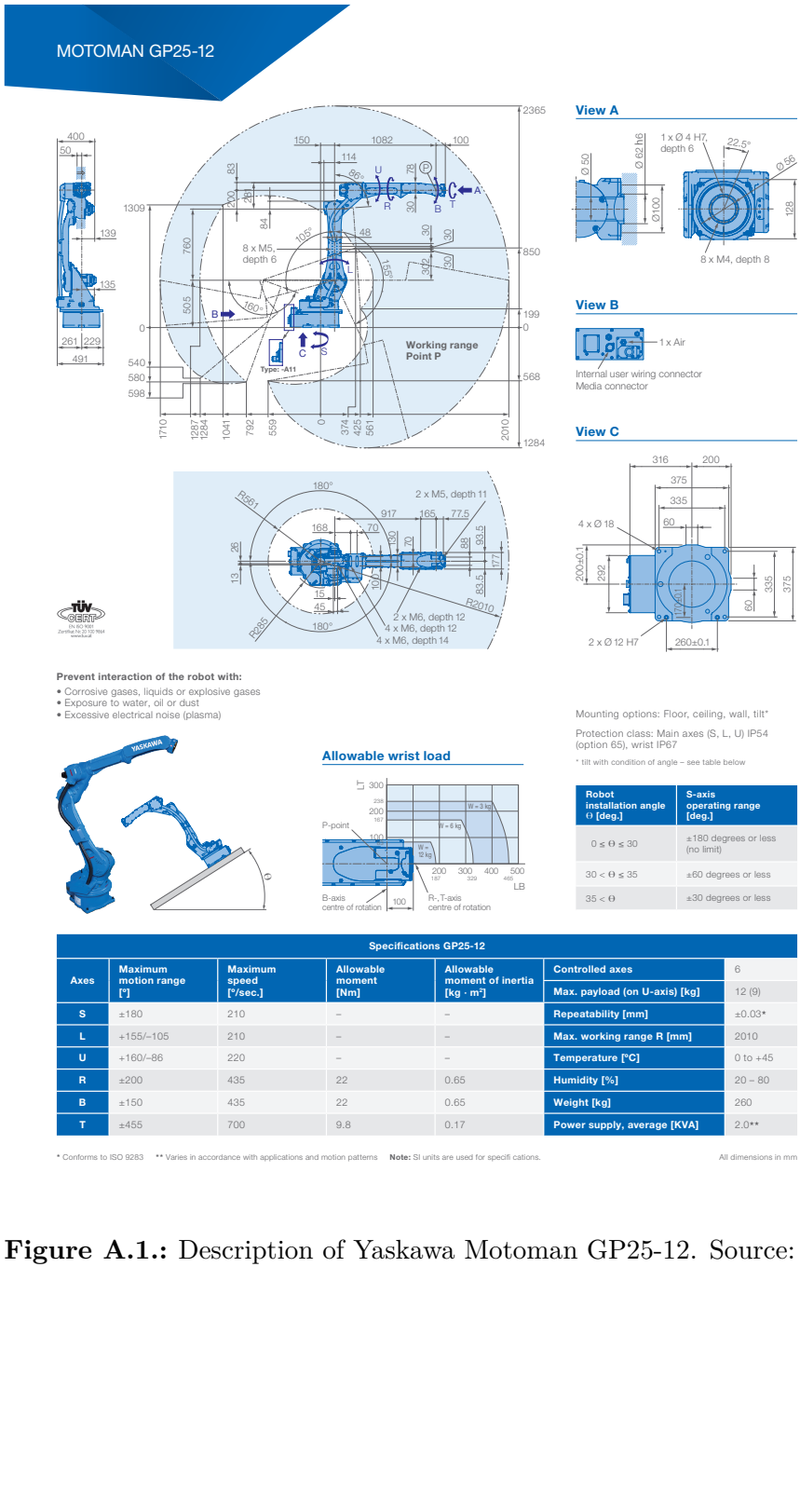
System description of welding cell

A.1. Fronius TPS 400i

Table A.1.: Technical data for TPS 400i. Source: [18]

Welding current max.	400 A
Welding current min.	3 A
Welding current / Duty cycle [10min/40°C]	400A / 40%
Welding current / Duty cycle [10min/40°C]	360A / 60%
Welding current / Duty cycle [10min/40°C]	320A / 100%
Operating voltage	14,2-34,0V
Open-circuit voltage	73 V
Mains frequency	50-60Hz
Mains voltage	3 x 400V
Mains fuse	35A
Dimension / b	300 mm
Dimension / l	706 mm
Weight	36,45 kg
Degree of protection	IP23

A.2. Yaskawa Motoman GP25-12



Robot installation angle Θ [deg.]	S-axis operating range [deg.]
0 ≤ Θ ≤ 30	±180 degrees or less (no limit)
30 < Θ ≤ 35	±60 degrees or less
35 < Θ	±30 degrees or less

Specifications GP25-12						
Axes	Maximum motion range [°]	Maximum speed [°/sec.]	Allowable moment [Nm]	Allowable moment of inertia [kg · m²]	Controlled axes	6
S	±180	210	–	–	Max. payload (on U-axis) [kg]	12 (9)
L	+155/–105	210	–	–	Repeatability [mm]	±0.03*
U	+160/–86	220	–	–	Max. working range R [mm]	2010
R	±200	435	22	0.65	Temperature [°C]	0 to +45
B	±150	435	22	0.65	Humidity [%]	20 – 80
T	±455	700	9.8	0.17	Weight [kg]	260
					Power supply, average [KVA]	2.0**

* Conforms to ISO 9283 ** Varies in accordance with applications and motion patterns **Note:** SI units are used for specifi cations. All dimensions in mm

Figure A.1.: Description of Yaskawa Motoman GP25-12. Source: [19]

Table A.2.: Specifications for Yaskawa Motoman GP25-12 part 1. Source: [19]

Axes	Maximum motion range [°]	Maximum speed [°/sec]	Allowable moment [N·m]	Allowable moment of inertia [kg·m^2]
S	± 180	210	-	-
L	+155/-105	210	-	-
U	+160/-86	220	-	-
R	± 200	435	22	0.65
B	± 150	435	22	0.65
T	± 455	700	9.8	0.17

Table A.3.: Specifications for Yaskawa Motoman GP25-12 part 2. Source: [19]

Controlled axes	6
Max. payload (on U-axis) [kg]	12(9)
Repeatability [mm]	± 0.03*
Max. working range R [mm]	2010
Temperature [°C]	0 to +45
Humidity [%]	20-80
Weight [kg]	260
Power supply, average [KVA]	2,0**

*Conforms to ISO 9283 **Varies in accordance with applications and motion patterns

Appendix B.

Python code

```
8
9   #Defining lengths [mm].
10  l1 = 505
11  l2 = 760
12  l3 = 200
13  l4 = 150
14  l5 = 275
15  l6 = 807
16  l7 = 100
17
18  #Values for x_tool, y_tool, and z_tool gathered from the programming pendant
19  x_tool = 0
20  y_tool = 450.27
21  z_tool = 84.4028
22
23  #M matrix for Yaskawa Motoman GP25-12 in zero-position without welding gun
24  M = np.array([[1,0,0,0],
25                [0,1,0,l4+l5+l6+l7],
26                [0,0,1,l1+l2+l3],
27                [0,0,0,1]])
28
29
30  #M matrix for Yaskawa Motoman GP25-12 in zero-position with welding gun
31  theta_endEffector = np.radians(-54)
32
33  R_endEffector = np.array([[1,0,0],
34                [0,np.cos(theta_endEffector ),-np.sin(theta_endEffector )],
35                [0,np.sin(theta_endEffector ), np.cos(theta_endEffector)]])
36
37  p_endEffector = np.array([x_tool,y_tool,z_tool])
38
39  M_tool = mr.RpToTrans(R_endEffector,p_endEffector)
40
```

Figure B.1.: Python code for calculating M matrix with and without tool

```
#Calculating each screw axis in space form Sn
w1 = np.array([0,0,1])
q1 = np.array([0,0,0])
v1 = np.cross(-w1,q1)
S1 = np.append(w1,v1)

w2 = np.array([1,0,0])
q2 = np.array([0,l4,l1])
v2 = np.cross(-w2,q2)
S2 = np.append(w2,v2)

w3 = np.array([1,0,0])
q3 = np.array([0,l4,l1+l2])
v3 = np.cross(-w3,q3)
S3 = np.append(w3,v3)

w4 = np.array([0,1,0])
q4 = np.array([0,l4+l5,l1+l2+l3])
v4 = np.cross(-w4,q4)
S4 = np.append(w4,v4)

w5 = np.array([1,0,0])
q5 = np.array([0,l4+l5+l6,l1+l2+l3])
v5 = np.cross(-w5,q5)
S5 = np.append(w5,v5)

w6 = np.array([0,1,0])
q6 = np.array([0,l4+l5+l6+l7,l1+l2+l3])
v6 = np.cross(-w6,q6)
S6 = np.append(w6,v6)

Slist = np.concatenate(([S1], [S2], [S3], [S4], [S5], [S6]), axis=0).T
```

Figure B.2.: Python code for calculating the screw axes in space form

```
#Calculating each screw axis in body form Bn
wb1 = np.array([0,0,1])
qb1 = np.array([0,-(l7+l6+l5+l4),-(l3+l2+l1)])
vb1 = np.cross(-wb1,qb1)
Sb1 = np.append(wb1,vb1)

wb2 = np.array([1,0,0])
qb2 = np.array([0,-(l7+l6+l5),-(l3+l2)])
vb2 = np.cross(-wb2,qb2)
Sb2 = np.append(wb2,vb2)

wb3 = np.array([1,0,0])
qb3 = np.array([0,-(l7+l6+l5),-l3])
vb3 = np.cross(-wb3,qb3)
Sb3 = np.append(wb3,vb3)

wb4 = np.array([0,1,0])
qb4 = np.array([0,-(l7+l6),0])
vb4 = np.cross(-wb4,qb4)
Sb4 = np.append(wb4,vb4)

wb5 = np.array([1,0,0])
qb5 = np.array([0,-l7,0])
vb5 = np.cross(-wb5,qb5)
Sb5 = np.append(wb5,vb5)

wb6 = np.array([0,1,0])
qb6 = np.array([0,0,0])
vb6 = np.cross(-wb6,qb6)
Sb6 = np.append(wb6,vb6)

Blist = np.concatenate(([Sb1], [Sb2], [Sb3], [Sb4], [Sb5], [Sb6]), axis=0).T
```

Figure B.3.: Python code for calculating the screw axes in body form

```python
#Defining each joint angle
theta1 = -np.pi/2
theta2 = -np.pi/5
theta3 = 0
theta4 = 0
theta5 = -np.pi/2
theta6 = 0

'''
Calculating the forward kinematics for the Yaskawa Motoman GP25-12
'''

thetalist = np.array([theta1,theta2,theta3,theta4,theta5, theta6])
T = mr.FKinSpace(M_tool,Slist,thetalist)
print('The configuration of the end-effector is given by the transformation matrix: ')
print(T)

'''
Calculating the inverse kinematics for the Yaskawa Motoman GP25-12
'''

#Defining the desired end-effector configuration to be equal to the solution of the forward kinematics
T_goal = T

#Defining a guess for each joint angle, as well as an positive error allowance
thetalist_guess = np.array([-1,-0.3,0,0,-1.2,0])
eomg = 0.001
ev = 0.0001

#Calculating the inverse kinematics for the Yaskawa Motoman GP25-12
[thetalist_0,result] = mr.IKinSpace(Slist,M_tool,T_goal,thetalist_guess,eomg, ev)
print("The angles required to obtain the desired end-effector configuration, T_goal, are:")
print(thetalist_0, result)
```

Figure B.4.: Python code for calculating the forward- and inverse kinematics

Appendix C.

Interview template

Robotic welding – Project thesis fall 2020

This is an anonymous questionnaire developed in order to establish various aspects regarding welding robots in the industry today. The results will be used in the subject TPK4650 - Robotics and automation, specialization project. The results will be made anonymous and will not be traceable to the one answering the questions. We want to conduct the questionnaire by first sending you this form, so you have time to think through the questions, then we want to conduct a small interview where we take notes of your answers.

1. How many years of experience do you have with welding robots?
2. How do you program welding robots?
3. Do you use sensors with your welding robots? If yes, which?
4. Which welding techniques do you use with your robots?
5. What is the biggest challenge you encounter while programming welding robots?
6. To what degree is one able to fully automate the welding process? Does one often have to complete the weld manually afterward?

Figure C.1.: Template sent to interviewees prior to interview