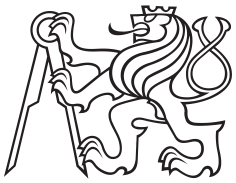
Master Thesis



Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Robotization of Front/Rear Window Operations during Final Car Assembly

Lucie Vajnerová

Supervisor: Prof. Ing. Václav Hlaváč, CSc.
Supervisor–specialist: Ladislav Kovařík
Study program: Cybernetics and Robotics
January 2024

MASTER‘S THESIS ASSIGNMENT

iii

I. Personal and study details

Student's name: **Vajnerová Lucie** Personal ID number: **483515**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Master’s thesis details

Master’s thesis title in English:

Robotization of Front/Rear Window Operations during Final Car Assembly

Master’s thesis title in Czech:

Robotizace operací s čelním/zadním oknem při finální montáži automobilu

Guidelines:

The work will be a case study motivated by operations on the final assembly line of the Toyota car manufacturer in Kolín on the assembly of the front and rear windows. The goal is to replace the current human operations with a force/torque-compliant robot. In the first part, the student analyzes the current solution and proposes a new one. It will take into account technical, economic, and ergonomic aspects. In the second part, he implements a demonstration task with a force/moment compliant manipulator when sticking tape on a window. Document the solution of both parts. Write your diploma thesis in English.

Bibliography / sources:

[1] Lihui Wang, Xi Vincent Wang, József Váncza, Zsolt Kemény: Advanced Human-Robot Collaboration in Manufacturing; Springer, Berlin; 2022
[2] Leong Wai Yie: Human Machine Collaboration and Interaction for Smart Manufacturing: Automation, Robotics, Sensing, Artificial Intelligence, 5g, Iots and Block chain, 2002, • ISBN: 97818395341402
[3] Uqba Othman, Erfu Yang Erfu Yang: Human–Robot Collaborations in Smart Manufacturing Environments: Review and Outlook, Sensors 2023, 23(12)

Name and workplace of master’s thesis supervisor:

prof. Ing. Václav Hlaváč, CSc. Department of Robotics and Machine Perception CIIRC

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **29.08.2023** Deadline for master's thesis submission: **09.01.2024**

Assignment valid until: **16.02.2025**

prof. Ing. Václav Hlaváč, CSc. Supervisor’s signature	prof. Ing. Tomáš Svoboda, Ph.D. Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

. Date of assignment receipt

Student’s signature

Acknowledgements

I want to thank my supervisor, Prof. Ing. Václav Hlaváč, CSc. for his guidance, help, and support throughout the course of this project. Next, I thank my supervisor-specialist, Mr. Ladislav Kovařík, for his assistance in this project at the Toyota car manufacturer in Kolín. I also thank Ing. Vladimír Smutný, Ph.D., Ing. Miroslav Uller, Ing. Pavel Krsek, Ph.D., and Ing. Libor Wagner from CIIRC CTU for their kind assistance and expertise.

Special thanks to my family, friends, and future cat, whose support and help were invaluable on my academic journey.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, January 9, 2024

.....

Lucie Vajnerová

Abstract

This diploma project contributes to automating three specific operations in the car final assembly shop at Toyota Motor Manufacturing Czech factory in Kolín. The thesis contributes by

1. *Proposing a robotized solution when preparing the rear and front windows* before they are installed into the car body. The work was conducted in a natural production plant environment, using Toyota procedures, because the diploma student has had partial employment there.
2. In a *university laboratory demonstration*, it implements a *sensory-motor behavior on a force/torque-compliant robot* Panda from Franka Emika company. The demonstration targets a particular industrial operation where the robot glues a rubber dam on the surface of a car window. Such an approach is novel to the factory. It serves as a proof-of-the-concept, enabling managers to evaluate the approach in the broader factory context and decide if to implement it on actual production lines in the factory.

Keywords: robotic automatization, force/torque control, panda robotic arm

Supervisor: Prof. Ing. Václav Hlaváč, CSc.
CIIRC CTU, Praha 6, Jugoslávských partyzánů 1580/3

Abstrakt

Tento diplomový projekt přispívá k automatizaci tří specifických operací na lince finální montáže v továrně Toyota Motor Manufacturing Czech v Kolíně. Tato práce přispívá

1. *Navrhnutím robotického řešení přípravy předního a zadního okna* před jejich instalací na automobilovou karoserii. Tato práce byla provedena v továrním prostředí a v souladu se zavedenými procedurami firmy Toyota. Studentka v továrně pracovala na částečný úvazek.
2. *Univerzitní laboratorní demonstrací*, kdy byl implementován *silově poddajný robot* Panda od firmy Franka Emika. Demonstrace úlohy se soustředí na jednu tovární operaci, při níž robot lepí gumové pásky/-přehrady na povrch okenního skla. Tento přístup je pro továrnu novotou. Slouží jako důkaz proveditelnosti, umožňující manažerům zhodnotit tento přístup v továrním kontextu a rozhodnout, zda implementovat tento přístup na produkční lince továrny.

Klíčová slova: robotická automatizace, silová poddajnost, panda robotická ruka

Překlad názvu: Robotizace operací s čelním/zadním oknem při finální montáži automobilu

Contents

Figures

Chapter 1

Introduction

1.1 Motivation

Automation is a process in which robotic solutions replace human-performed tasks. Automation is a powerful tool for optimizing car manufacturing processes in the automotive industry. Automated processes are faster, more reliable, and provide better accuracy than human operators performing tasks. They can enhance and streamline production lines in various industries like production, manufacturing, engineering, automobile, medical, or aerospace [6].

This work is related to a factory plant producing cars, more specifically assembly processes on rear/front window assembly at Toyota Motor Manufacturing Czech (TMMCZ) in Ovčáry near Kolín. The author of this work had been working there part time since May 15th, 2023.

Automotive plant consists of several "shops". First is the forming shop, where presses form the sheet metal into various panels. The second is the welding or body shop, where formed sheet metals are welded together, and bare car bodies are produced. Robots perform most operations in welding shops. They weld, handle material, and apply adhesives and coating against corrosion. The next stop is the paint shop. It includes cleaning, priming the car body, and spray painting it. Robots here apply painting and ensure its smoothness. Finally, the painted car body is brought to the assembly shop. Here, car parts such as chassis and engines, wiring systems, wheels, doors, windows, seats, and more are assembled. Robots on final assembly lines

are able to manipulate heavy parts and handle many different task-specific operations [7].

Final assembly lines in the automotive industry require more human operators than other shops due to their requirement for flexibility and robustness. Assembly processes are skill-based operations that require procedural skill, where operators execute tasks in a predefined order [8]. As such, automation is rarer in the final assembly and presents interesting technological challenges.

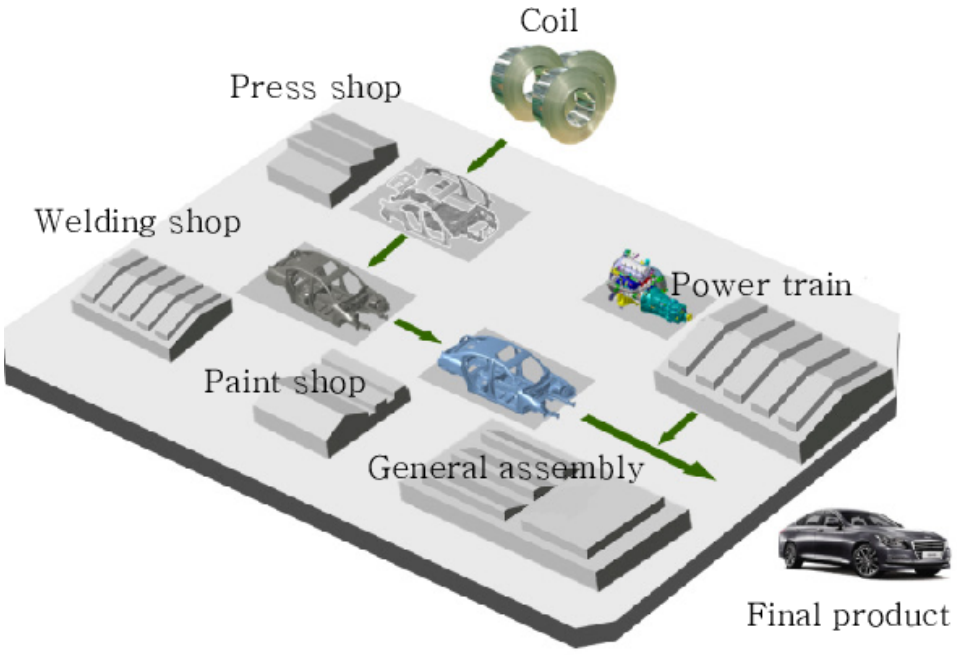


Figure 1.1: Automotive manufacturing process [1].

The motivation for this thesis was to automate three final assembly processes on rear/front window assembly at TMMCZ plant. Toyota Motor Corporation (TMC) is a Japanese company that designs, manufactures, assembles, and sells passenger cars, minivans and commercial vehicles [9]. Toyota has distinctive competence in its production system thanks to the "Toyota Production System". This concept involves innovative practices like Just in Time (JIT), Kaizen, or Six Sigma that help improve production time and optimize manufacturing [10][11]. TMMCZ manufacturing plant was opened in 2005. Initially, it operated as a joint venture between TMC and PSA Peugeot Citroën. In 2021, TMC became the sole owner of the factory [12], [13]. In 2023, TMMCZ manufactured two types of cars – Toyota Aygo X and Toyota Yaris. Both cars have relatively short car bodies. TMMCZ has uniquely adapted its production lines to allow for faster production.

At TMMCZ, any suggestions for improvements or new automation of pro-

duction lines undergo intensive planning. At the beginning of this process, multiple solutions are considered from technical, technological, safety, economical, and logistical standpoints. These solutions are presented to managers at the company, who choose a single solution, if any. Next, a so-called "specification" must be created. Such document presents all necessary information and requirements to be submitted to several companies that can subsequently attempt to be selected as providers for the project. After a company is chosen, the project can finally be implemented and/or built at the TMMCZ factory.

1.2 Task Specification

The diploma project assignment requires a case study of the preparation of front/rear windows to be mounted onto a car body at the final assembly line in TMMCZ. The goal is to replace the current human operator with a force/torque-compliant robot.

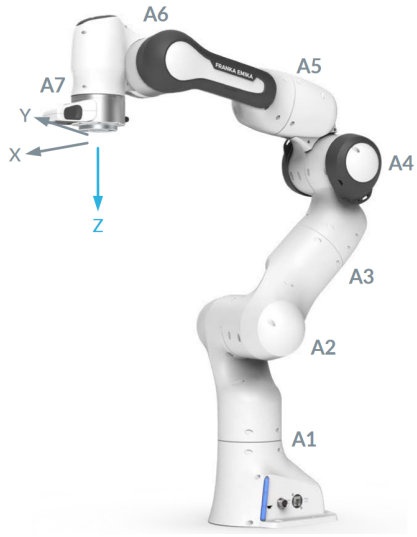


Figure 1.2: Panda robotic arm from Franka Emika [2].

The force/torque-compliant robots have not been implemented in TMMCZ yet. The TMMCZ management is not against the further introduction of such robots into their assembly line, but for the moment asked the author of this thesis to consider standard robots in her analysis and design as well.

It was specified in the assignment that the thesis would have two parts.

Part 1 – Robotizing car front and rear windows operations.
The student has to analyze the current solution and propose a new one, considering technical, economy, and ergonomic aspects. The new solution should replace a human worker with a robot.

Part 2 – Demonstrating the use of a force/torque compliant robot.
CIIRC CTU lab has appropriate force/torque-compliant robots. The Panda robot from the German company Franka Emika is available and was recommended by a supervisor to the diploma student, Fig. 1.2. The assignment is to use this robot to glue rubber dams to the car window in a university lab. The outcome should be a demonstration of the task.

1.3 Thesis Organization

In Chapter 2, we overview the related works and present logistical, economical, safety, and ergonomic definitions which we must consider as part of our work. We also compare collaborative robots (cobots) with industrial, and provide reasoning for our choice.

In chapter 3, we seek to automate the three current processes on the rear/front window production line. We explain why certain materials and procedures are used and their role in the final assembly.

Chapter 4 contains our proposals and solutions to automating front and rear window preparation operations on the final assembly production line.

Chapters 5 and 6 discuss our demonstration task using a force/torque-compliant robot in the university laboratory conditions. They explain the theory behind our code and try to follow the commonly used pipeline of robot planning used by robotic companies. We also introduce the software tools used for our final demonstration.

Chapter 7 summarizes achieved results, evaluates them critically, and establishes possible future work.

The Appendix, Chapter A, provides designs of our end effector, tables of financial calculations, a project report and a demonstration video.

Chapter 2

Overview of Concepts

The term "Industry 4.0" has become a leading direction in making industrial production more effective since 2013. The term digitization of industry is used for the same thing outside of Europe. The term refers to a new industrial revolution integrating digital technologies into industrial processes, i.e., communication, modeling (digital twinning), and production planning. The products are designed, produced, and delivered with advanced technologies to create "intelligent factories" [14]. Furthermore, Industry 4.0 also focuses on the close working relationship between human operators and robots, as well as the effortless adaptability of already implemented technologies, e.g., for new models of cars and new technologies like smart sensors, programmable devices such as PLC / HMI, and industrial or collaborative robots [15].

We present an automation solution for window preparation assembly line processes. This chapter introduces important factors we must consider as part of our offered solutions: logistics, economic factors, safety, and ergonomics.

2.1 Industrial and Collaborative Robots

We deal with a subset of robots, i.e., industrial manipulator arms. Such a robot is a mechanically jointed structure of different configurations. Joints are either revolute (R) joints with rotational motion around an axis, or prismatic (P), which provide linear motion along an axis. Joints are combined serially. The robot configuration is often briefly characterized as, e.g., RRR, RRP for

a three-joint robot [16].

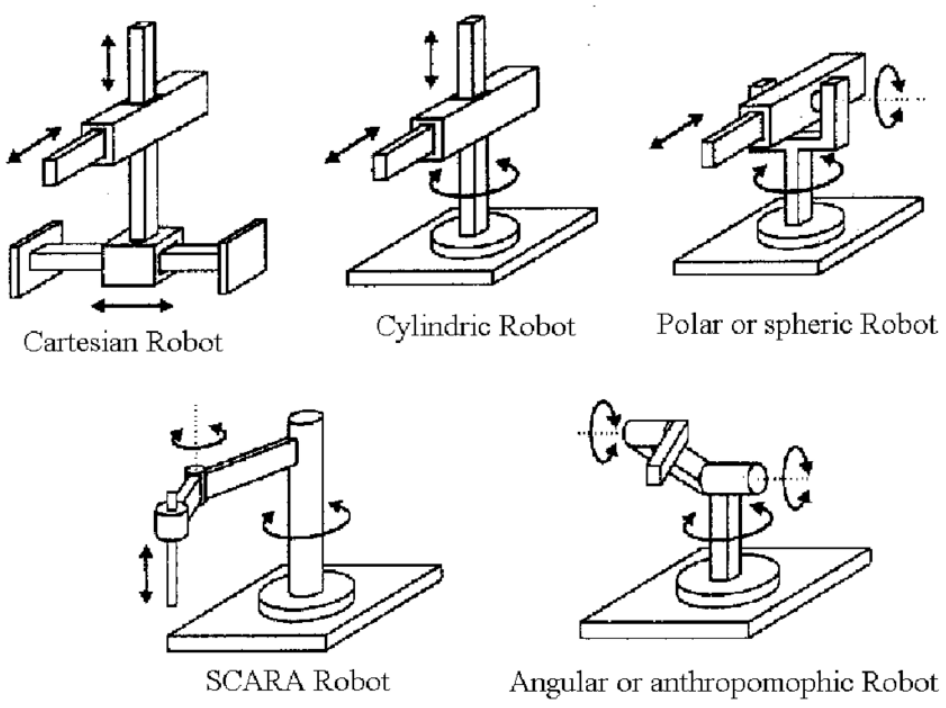


Figure 2.1: Common industrial robot arm geometries [3].

There are five major mechanical configurations commonly used for industrial robots: Cartesian (PPP), cylindrical (RPP), spherical (RRP), articulated (RRR), and SCARA - Selective Compliance Articulated Robot for Assembly (RRP). You can see them in figure 2.1. Robots are also characterized by "Degrees of Freedom" (DoF). These independent joints can provide freedom of movement for a given manipulator. The Panda robot used in our demonstration task is a 7 DoF, fully articulated robotic arm.

The higher requirements the world has for robotic applications, the more flexible and variable the robots need to become. This is especially important for industries with shorter product cycle lifetimes and frequent production changes. In the recent years, industrial robots have become less desirable due to often being overly single-specific-task oriented. Human-Robot Collaboration (HRC) is the preferred solution and current trend. HRC allows for higher levels of productivity, quality, and flexibility. The robot performs heavy-duty tasks, monotonous operations, or hazardous implementations, which allows more intricate and innovative operations to be handled by the human worker. This optimization of tasks can improve manufacturing time and quality.

In Industry 4.0, smart manufacturing became more popular. Various techniques, such as combining Artificial Intelligence or Augmented reality

into cobot operations or creating a Digital Twin, can be used to improve production. While everyday industrial robots still outweigh the number of cobots in the automotive industry, thanks to cobots' better flexibility and compact space requirements, we can expect their numbers to rise.

2.2 Logistics

The automotive industry is experiencing an increase in the complexity of customized models and variants of cars. For example, TMMCZ produces two types of cars – Toyota Yaris and Toyota Aygo X. Customers can customize their car from predefined models and options. To keep customization open, but still ensure low production time, automotive manufacturers standardize construction models into common platforms. This approach reduces costs by sharing components. These changes necessitate flexible and agile automotive supply chain management, impacting assembly operations and logistics [17].

Parts logistics must ensure timely deliveries to exact operator task spaces so that the final assembly lines never run out of parts and all parts correspond to the correct car order on the rack [18]. Failure to do so can cause factory line stops and incorrect car assemblies and lead to massive financial losses. However, large stock at the production stations leads to high handling and holding costs.

The Toyota Production System works on a Just-in-Time (JIT) system [19]. JIT is also a management philosophy that seeks to eliminate all excessive "waste" in both the general sense and time or resources. In-house, this means the existence of a so-called "supermarket-concept", where each supermarket is a storage facility in the direct vicinity of the final assembly production stations. It serves as an intermediary store for parts. Tow trains (towing vehicles with a handful of wagons driven by human operators) set off at predetermined schedules along given routes and supply stations with new parts or collect the used packaging. This presents another optimization problem: the number and placement of supermarkets on the shop floor. Striking a balance is crucial, as an excessive number may incur more costs than benefits, while too few or poorly placed supermarkets may undermine their positive effects.

2.3 Economy

Before automation can be implemented, the financial side of things must be considered. First, let's look at the viewpoint of the company. Changing platforms and upgrading technologies is expensive and may be therefore challenging for the company's budget. The initial investment in robots must not outweigh the wage of human operators within the given time frame, during which we consider the return on investment. This time frame is usually considered in a matter of years. Therefore, it must be carefully analyzed if automation is worth it.

Secondly, fears of losing employment by replacing human operators with automation solutions should be considered. TMC tries to ensure no worker displacement happens due to automation or digitization. Instead, workers are re-trained for new tasks, reassigned to other posts, or re-skilled. Research conducted by the International Labor Organization on the automotive sector in Mexico (including the local plant of Toyota Guanajuato) concludes that "the arrival of new models and the implementation of new technologies expand business without reducing employment" as the factories choose to retrain workers and improve their digital skills [20]. A similar result has been reached in a discussion paper by the British Centre for Economic Performance, which claims that their results "indicat[e] that in practice the productivity effect tends to outweigh the displacement effects" [21]. On the other hand, research conducted on automation's impact on general income argues that automation leads to wealth and overall income inequality [22].

2.4 Safety

Collaborative robots must be safe for their human operators. Interactions between cobots and humans can be sorted into different categories. The "coexistence" without physical barriers. In the "synchronization" interaction, the human operator and robots share a workspace, but perform their tasks sequentially at different times. "Cooperation" is defined as a human operator and a robot working simultaneously towards a shared objective while having separate interests. And lastly, "collaboration" is when a human operator and a robot work together towards a single goal in the same workspace at the same time [23].

Cobots are defined by their safety. Four standard safety approaches exist:

(A) an alert for the operator if they get too close to the robot; (B) a full stop of the robot when sensors detect human presence; (C) moving the robot away from where detection systems report unexpected movement; and (D) modifying the robot's trajectory away from the operator in real-time monitoring the human's behavior within robot workspace and prediction of their future behavior [24].

Industrial robot legislature is fairly complex and formally defined. On the other hand, cobots' safety rules are still somewhat ambiguous as their rapid development precedes its legislature. Currently, the only statement in place of cobots' safety is ISO/TS 15066, "Collaborative Robot Safety", which is an international standard [25]. It defines four modes of safety approaches and seeks to define "pain thresholds" for robot-human incidents to guide robotic workplace design. Of course, even if a cobot is deemed "safe" by the standard, any modifications of the cobot (including installation of a new gripper or a cobot holding a new part) require extensive consideration to determine if the robot will still be safe under such new parameters.

2.5 Ergonomics

Ergonomics considers the relationship between human operator and their workspace. It considers risk factors like repetitive motions, static posture, or heavy lifting while trying to determine their possible effects on worker safety and health [26]. Motions that, at first glance, are not difficult to perform can cause strain to soft tissues like muscles, tendons, or ligaments over a longer period of time spent repeating those motions. This can expose workers to Musculoskeletal Disorders (MSDs), which can eventually escalate to movement disabilities, muscle loss, paralysis, and in the worst case - death. It is, therefore, extremely important to perform an ergonomics assessment on every workstation.

Rapid Upper Limb Assessment (RULA) is a popular survey method for investigating ergonomics within workspaces that pose a risk of MSD. It was first introduced in 1993 by L. McAtamney and E.N. Corlett [4]. RULA uses body posture diagrams and three different scoring tables. It divides the human body into two groups: Group A for upper arm, lower, and wrist position (see below 2.2), and Group B for neck and trunk analysis. The first table determines Group A's score, the second table records score for Group B, and the third table shows final RULA score by combining information from the first two tables. While RULA is still used today, there are limitations to its method. First, the left and right sides are both considered separately;

it does not take into account single fingers or thumb movement; and RULA also only considers static posture.

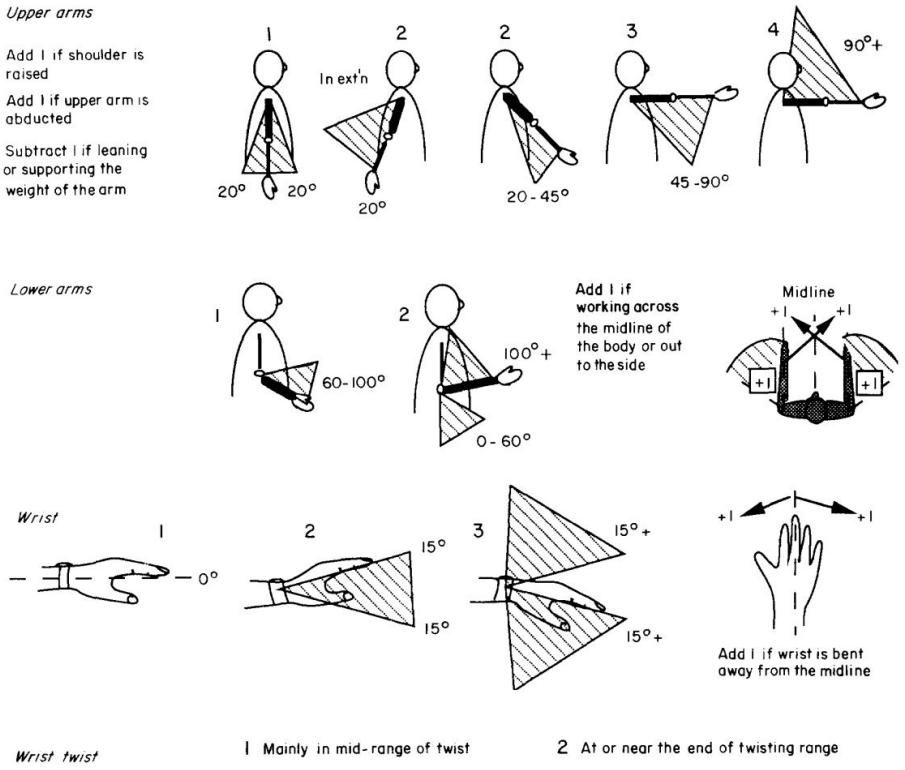


Figure 2.2: RULA posture scores for body part group A [4].

Rapid Entire Body Assessment (REBA) is another popular method. RULA and REBA share some principles for evaluating risk assessment and designing body parts. However, REBA focuses on different professional fields, namely health care and service industries. REBA evaluates the whole body, while RULA focuses on upper-body movements. Furthermore, REBA allows for dynamic posture evaluation. Despite this, REBA is not ideal for evaluating production line ergonomics.

Chapter 3

Our Tasks and Processes

The thesis offers solutions to automate three processes in the Final Assembly shop at TMMCZ. The front/rear window shield preparation phase was our focus. These processes are termed F108, F109, and F116 in the factory. Each process must follow the factory-wide set cycle time of 72 seconds. In other words, all processes must take 72 seconds in average. This includes processes in Final 1 - a section of the final assembly shop that sets windows, car seats, and more. In-house, the two different models of cars are given abbreviations codes: G1 denotes Toyota Yaris, G3 denotes Toyota Aygo X. Moreover, the front window is sometimes called simply "FR" and the rear window as "RR". Dimensions of windows can be seen in Table 3.1. The smallest window is G1's RR, and the largest is G1's FR.

	G1 (Yaris) [mm]	G3 (Aygo X) [mm]
FR	1000x1400	800x1300
RR	450x1300	700x1050

Table 3.1: Window dimensions (values are approximated by the maximum in the respective direction).

Figure 3.2 depicts the window line as part of the Final 1 layout (seen in Figure 3.1). The window line is located on the left side of the car hanger line at the very beginning of the Final 1. The car hanger lowers cars from a car rack above the factory's ground floor. The cars are lowered next to the window line slanted at 15 degrees until they reach the ground level. The Window line is supplied with windows from an automated loader. Smaller parts are delivered through a supply system under the car hanger as it leaves the feeder. At the end of the window line, an automated robotic solution applies urethane glue on the window. Finally, after this operation, the window can be glued to the car body. The only possible way to enter the window

line is by walking through the Final 1 assembly line beneath the car hanger. All layouts in this thesis keep to a measurement system of 100×100mm per single square.

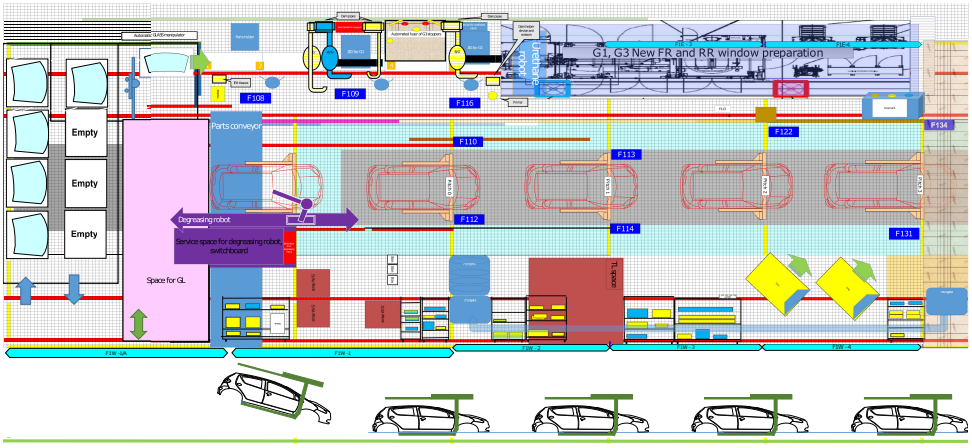


Figure 3.1: Start of Final 1 and its layout.

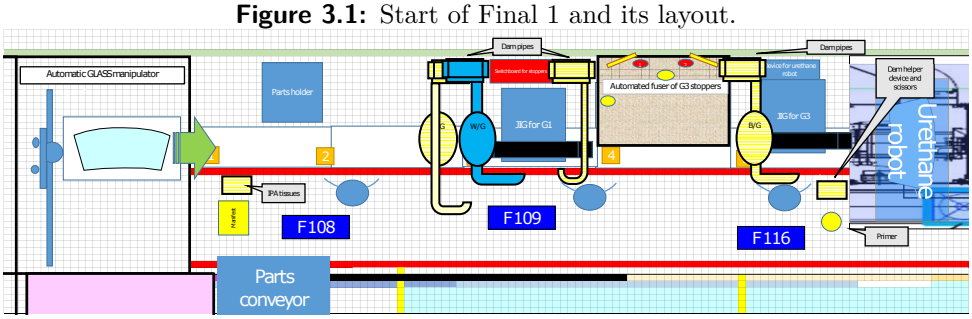


Figure 3.2: Zoom-in on window line layout.

A human operator performs each process for one car's window pair (FR/RR). For example, if the current car model is G1, the operator receives G1's FR first. She/he performs all necessary tasks of their process before passing the window to the subsequent process. Then, the same operator takes G1's RR, performs similar tasks as before (adapted for RR), and then passes the window to the following process. This order of tasks for a single pair of windows is what we mean by a process.

Generally, if we looked at a single window passing through the window preparation line, its order of tasks would look as such:

1. Automatic loader of windows passes a window to the window assembly line to process F108.
2. F108 operator takes the window and cleans its edges with a wet wipe.
3. If the current window type is FR then the F108 operator assembles and mounts parts such as a mirror, camera, and rain sensor to the window.

4. F108 operator passes the window to F109 station.
5. At F109 station, the local operator takes the window and binds stoppers to the appropriate markers. In some cases, an automatic stopper heater machine binds stoppers instead of the operator.
6. F109 operator glues the set of rubber dams on the window (exact placement depends on window type and model).
7. Then, F109 operator passes the window to F116 station.
8. F116 station operator takes the window and applies all remaining dams on their required positions.
9. F116 operator applies primer.
10. The window passes from station F116 to the automated urethane application robot.

We can separate the three processes (F108, F109, and F116) into five distinct tasks. These are

- Cleaning with isopropyl alcohol (IPA)
- Mounting parts (mirror, camera, rain sensors etc.)
- Binding stoppers
- Gluing rubber dams
- Applying the primer

We can discuss each of these tasks in more detail now.

3.1 Cleaning with IPA

Isopropyl alcohol is an organic disinfectant/cleaner often used for cleaning purposes. Typically used in solutions with 60-90% water, it can kill up to 99.99% of germs within 10-30 seconds of its application [27]. In TMMCZ in process F108, IPA is used to clean the edges of windows to prepare them for further application of rubber dams and urethane. The windows are delivered

by an outside manufacturer and delivered to the TMMCZ factory. During transportation, windows can become contaminated, and this can cause issues when gluing dams or applying primer.

At F108, apertured wipes with 99% solvent of IPA are used for cleaning. One wipe tissue is used and discarded for each pair of front/rear windows. The time of this task as part of F108 is set at 14 seconds for Yaris and 15 seconds for Aygo X. Visualization of these trajectories can be seen in 3.3.

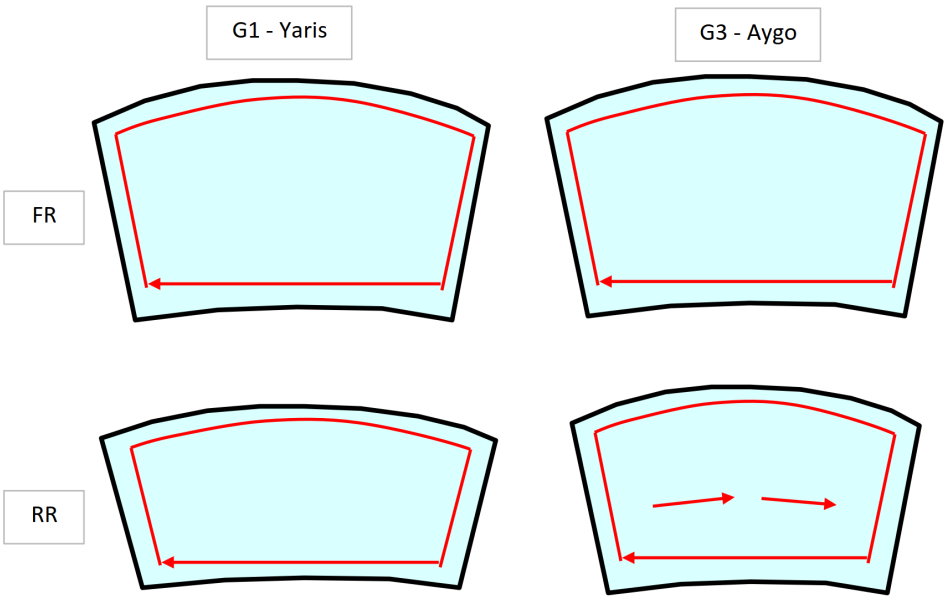


Figure 3.3: Visualization of IPA cleaning locations on windows Toyota Yaris and Toyota Aygo X.

There are not many works of others related to this task. Most modern "window cleaners" are used for cleaning high-rise buildings or domestic use [28]. Cleaning high-rise building windows can be highly hazardous. It becomes safer by using robots instead. The cleaning mechanism usually follows one of two principles: drag-wiper or roller-wiper. Research in [29] compared these two principles with force analysis, energy consumption, and wiping effect. They concluded that roller wipers have a better cleaning effect and energy consumption. An alternative to the roller-wiper can also be the water-spraying technique. Finally, sensors are essential for determining the robot's position and internal and external status of the robot.

A recently implemented automation similar to IPA cleaning already exists in TMMCZ. Bc. Suyash Kashyap's bachelor thesis already describes this approach [5]. The thesis is entitled "Automated degreasing application for car window body flange" and presents designs for the automation of degreasing/wiping car body flange process. Kayshap's designs served as base

designs for real equipment 3.4. It uses a pair of two cobots (one for the front and one for the rear side of the car body) and an ultrasonic cutting station for wipes. Wet wipes are picked up from cutting stations by cobots' soft grippers. Then, cobots follow a learned trajectory along the car body flange, cleaning it using roller-wiper movement. Ultimately, cobots return to their starting positions and discard used wipes. As this process is done on the car rack, the cobots are mounted on a linear track that moves the cobots in synchronization with the car rack movement.

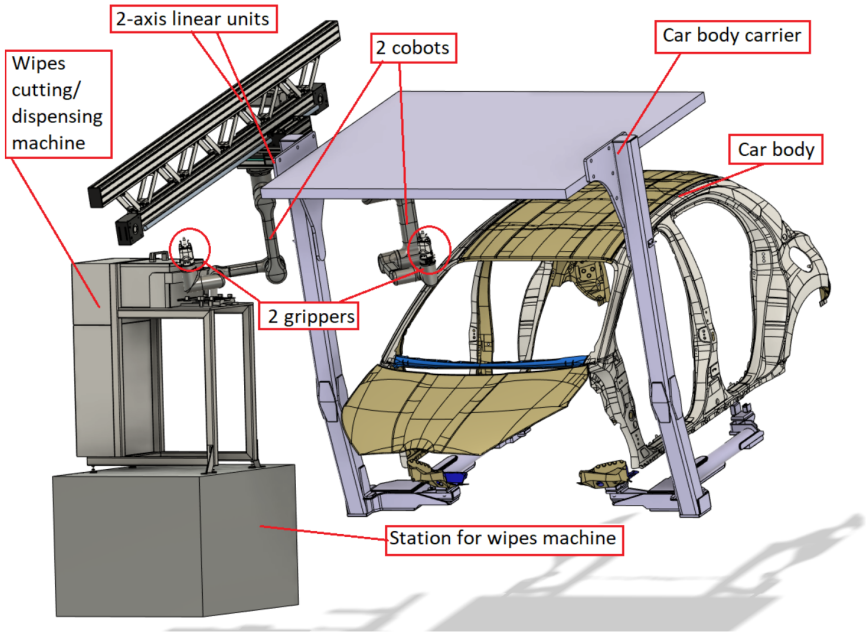


Figure 3.4: Final solution for window body flange degreasing project by Bc. Suyash Kashyap [5].

3.2 Mounting Parts

Modern cars have rear-view mirrors, cameras, and rain sensors on the top portion of their front window. Yaris and Aygo X are no exception. These parts are mounted to the FR in process F108 at TMMCZ. The order of tasks is as follows:

1. The camera's heating and cable must be assembled on the window into a pre-manufactured case.
2. The camera is mounted on the case and then its safety.
3. Rains sensors are pressed into the case next to the camera.

4. The rear-view mirror is mounted on top of the case using a tightening drill.

5. Operator ensures everything is correctly mounted. If yes, he passes the window to position F109.

These tasks take 34 seconds for both G1 and G3. The order of tasks for this is given and cannot be changed (parts strictly fall into the case). This task is the most "manual assembly" detailed task on the window preparation line. It requires precise use of multiple operations: cable manipulation and placement in tight quarters, tightening of screws in limited space and gentle pressing of sensors into cases until they "snap in".

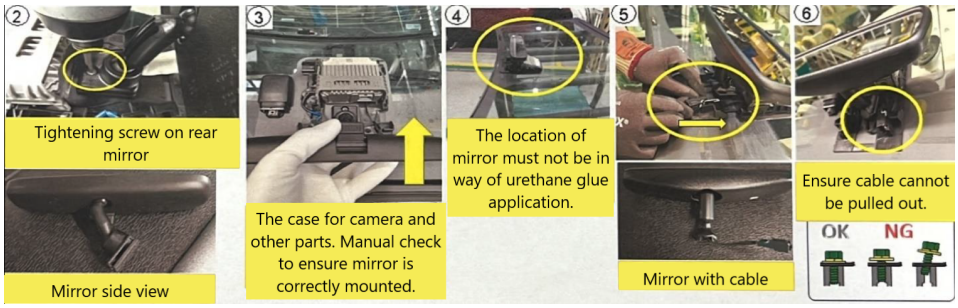


Figure 3.5: Preview of rear mirror mounting task and its product checking process.

Cable manipulation carries several difficulties for robotic manipulation [30]. Firstly, cables are considered a deformable object. To work with deformable objects, multiple things must be considered: gripper and robot design, sensing, modeling, planning, and control. Robot sensing especially plays a vital part - deformable objects must be sensed at all states to be able to plan control for them. Standard sensing techniques include visual, tactile, and force sensing. Visual is used most often for rope/wire sensing. However, the cable can get partially occluded. Various methods to track obscure cables exist. For example, in [31], a combination of YOLO (a real-time object detection algorithm) and a traversing algorithm (a tree data structure search) is used for a cable untangling problem in a constrained space. The paper concludes that such a method could find use in the automotive industry. More sensors can be introduced to improve cable tracking. Tactile grippers are suitable for cable manipulation as well [32]. However, such grippers are often large and would not fit inside, say, a case for mounting a camera on the front window shield.

Tightening screws also present interesting challenges [33]. First, it's necessary to fit a screwdriver tip into the drive of a screw. Second, a correct torque must be applied for the screw to be tightened correctly. Alignment of parts

can be done using vision sensors or force and position feedback. The fitting of the screw into a correct hole can be seen as a common robotic task of peg-in-hole. However, due to the need to fit the screw into the correct position with its external thread, it is challenging to implement well-known approaches to general peg-in-hole tasks. Moreover, a hybrid controller should be implemented for screw driving that integrates position, force, and admittance controls.

3.3 Binding Stoppers

Stoppers are put on windows to "fix" them into the correct positions on car bodies. At TMMCZ, outside manufacturers have already glued some stoppers on the window. Other stoppers need to be applied to the window in-house. For each car model at TMMCZ, stoppers look differently.

For G1, stoppers are small, hard, plastic rectangular parts of around 2cm in length. They have protective tape on the bottom, which needs to be removed to reveal the glue. Then, the stoppers are placed on marks on the left and right sides of the G1 window around their midpoints. This task takes 10 seconds and is part of F108.

On the other hand, G3 requires its stoppers to be heated onto the windows using a specialized automatic stopper welding machine. A glue is applied to the window on stopper positions, and then the window is passed into the welding machine. This machine works on the principle of ultrasonic welding, where high frequency produces low-amplitude mechanical vibrations. Parts join due to heat generated by friction and plastic parts' deformation [34]. This task must be manually set up beforehand by the operator. It takes 11 seconds and is part of F109.

3.4 Gluing Rubber Dams

Before adhesives are applied to the window to glue it to the car body, TMMCZ first glues rubber dams on windows. These long rubber strips of 5×5mm width and height prevent adhesive glue from leaking into undesired places. They act as a spacer. Such technique has long been a part of gluing windows on car bodies, even in the 20th century [35]. The rubber has good flexibility

and can be bent around window corners. It also helps distribute stress on the window in cohesion with urethane adhesive, giving good buffering qualities.

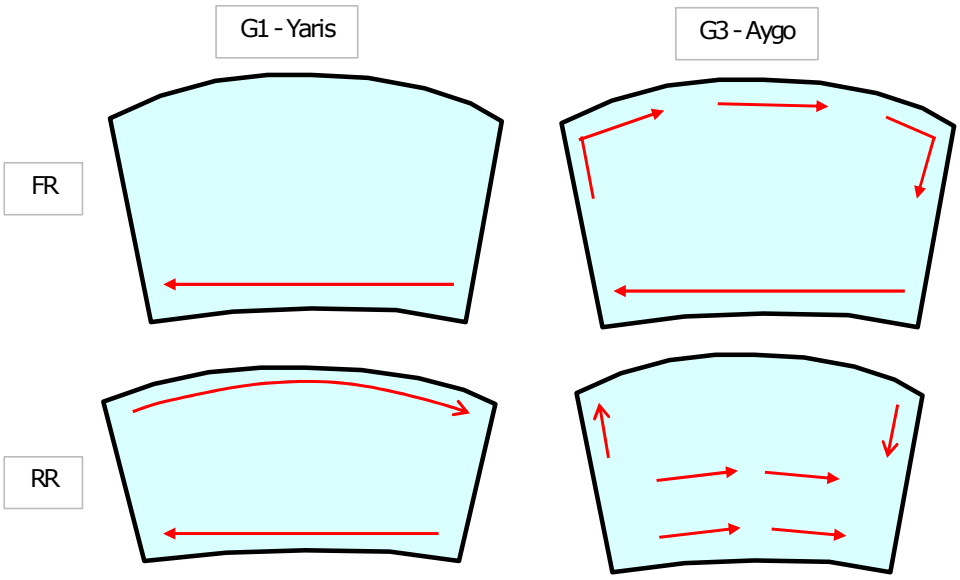
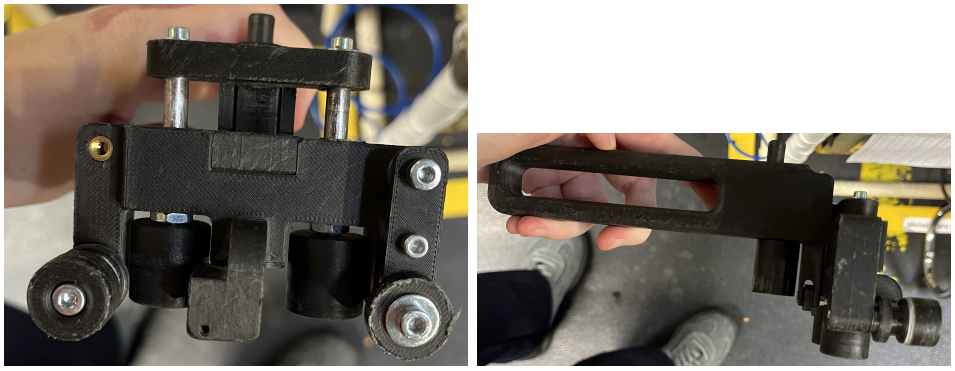


Figure 3.6: Visualization of dam gluing locations on windows Toyota Yaris and Toyota Aygo X.

At TMMCZ, dams are glued on windows manually. This process takes 36 seconds for G1 and 78 seconds for G3 and happens on both F109 and F116. G3 requires more time due to its RR and four strips of dam glued on its middle, Figure 3.6. A particular helping jig, Figure 3.7, is used by the operator to glue bigger portions of dams and ensure correct positioning of dams, while others are cut beforehand by an automatic dam cutter to predefined sizes and then glued on cars by hand (RR G3 specifically). Bigger special jigs that fit on the bottom side of FR windows are used to ensure correct dam positioning on FR of both car models.



(a) : Front view **(b) :** Side view.

Figure 3.7: Manual dam gluing jig (a helper device).

Generally, gluing tapes on various materials and shape types is a known

robotics problem for many tasks and processes (for spray painting, plasma spraying, etc.) [36]. A good surface covering strategy and proper tape attachment from the end effector must be guaranteed to ensure we tape correctly. There are multiple options for designing taping end effectors. 3D scanning methods can help in cases where material has an irregular shape. We can move to path planning once the object's shape is obtained. We must then design a proper taping end effector and program and design a robotic platform for such a task. To correctly glue tape on the object, the taping tool/end effector can apply a certain force on the object. Compliant robot behavior helps with such tasks and force feedback can be implemented to improve the tapping task.

3.5 Applying Primer

Primer is part of epoxy, a structural adhesive that helps bond different types of surfaces to one another [37]. The epoxy comes in two parts - a resin and a catalyst. The catalyst helps the resin harden and bond in their combination. A common epoxy type (that is used at TMMCZ as well) is a polyurethane epoxy. They are more flexible than other epoxy types, but can break by shearing. On the other hand, their ability to absorb forces and transmit them more evenly across their surface can help improve the safety of drivers and are thus desirable in the automotive industry (rather than welding windows on car bodies). Urethane cures by air moisture, meaning that curing time can vary depending on factory location. Curing times can be measured in minutes or hours.

TMMCZ uses a black, thin polyurethane primer as an adhesion promoter on glass for bonding with polyurethane sealants. Its drying time is slightly over 5 minutes, and its service temperature is -40 to 90°C [38]. The surface we want to apply primer to should be dry, free of oil or grease, and generally clean.

This task takes effect on process F116, taking 33s for G1 and 34s for G3 windows. This task must come last, as right after the primer application, the windows are given to the urethane robot that finishes the resin application so that windows can be glued to the car body. The primer is applied mainly on the inner portion of rubber dams. The locations can be seen in figure 3.8.

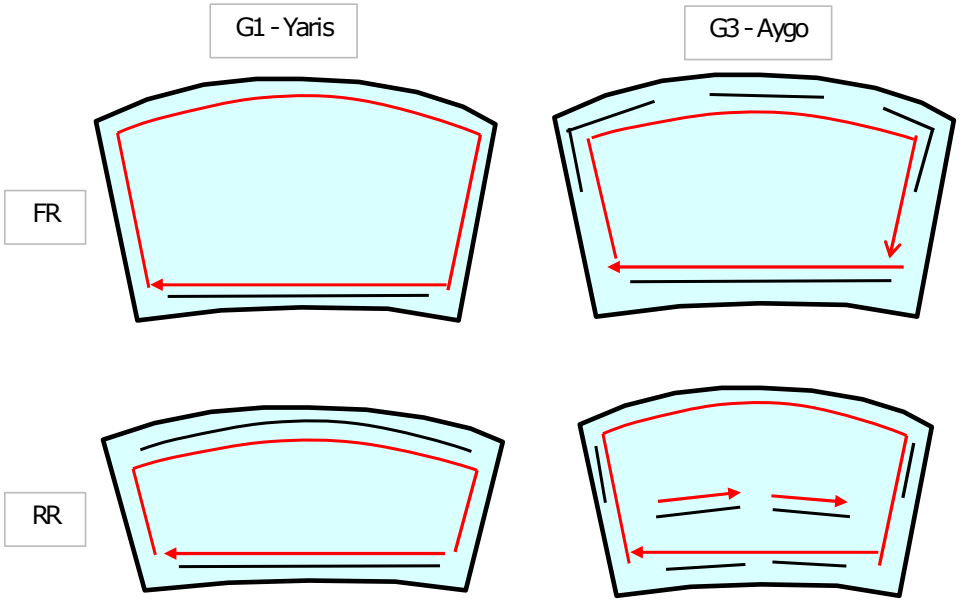


Figure 3.8: Visualization of primer application locations on the windows of Toyota Yaris and Toyota Aygo X.

Chapter 4

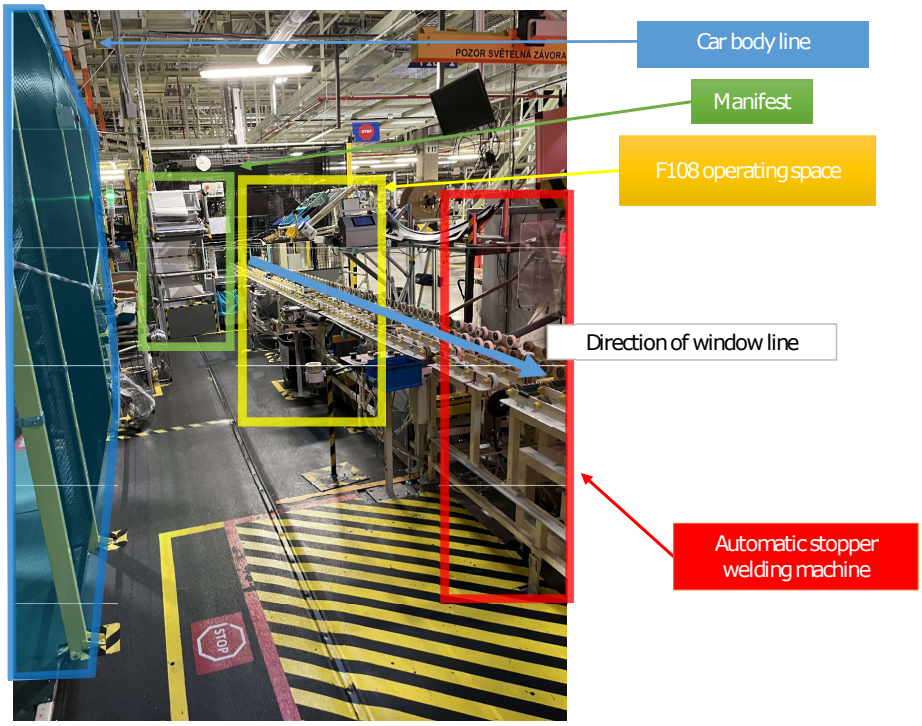
Proposed Automation Solutions

The window line at TMMCZ has a total of three processes we seek to automate (processes were described in Chapter 3). The current line layout can be seen in figure 3.2. Photos taken from process position F109 can be seen in Figure 4.1 for better visualization.

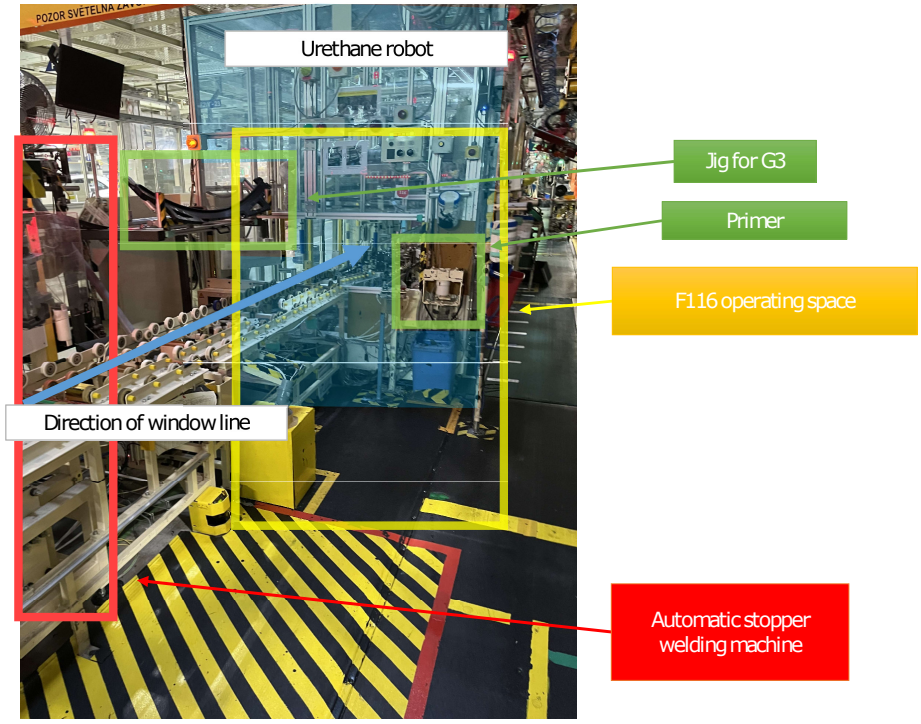
4.1 Automation of Tasks

As discussed in previous chapters, we have five tasks we can possibly automate on the window preparation line. Three of these tasks are viable for automation when considering technological, ergonomic, and financial aspects. The tasks we would suggest to automate should also be "generic" enough that robots could quickly adapt to new window shapes with a new car model (i.e., not G1 or G3). As such, we will presume that new car models will still be glued on the robot using the same polyurethane adhesive that requires primer application and dams.

What follows is a more detailed analysis of three tasks not recommended (or only in some cases) for automation:



(a) : Left side view.



(b) : Right side view.

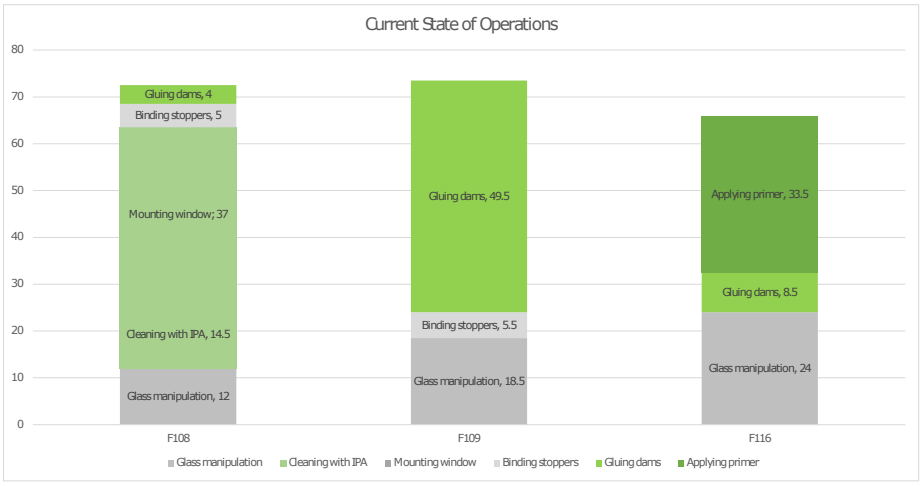
Figure 4.1: Photos of window preparation line from position F109 at Final 1.

Cleaning with IPA: The task of cleaning windows with IPA takes an average of 14.5s. It can be considered an "easy to automate" task (more on below). If cleaning windows were left up to the operator, the total time of our newly suggested process would be 73.5s on average. This is slightly above the cycle time. However, that is also the current process time for F109. As such, it is possible, given TMMCZ standards. We must then consider if, from a financial perspective, it is "worth it" to automate this task. On the other hand, having the robot clean the window would provide better ergonomics for the operator process, save time for operator tasks, and (as we will see below) provide safer alternatives for Final 1 layouts.

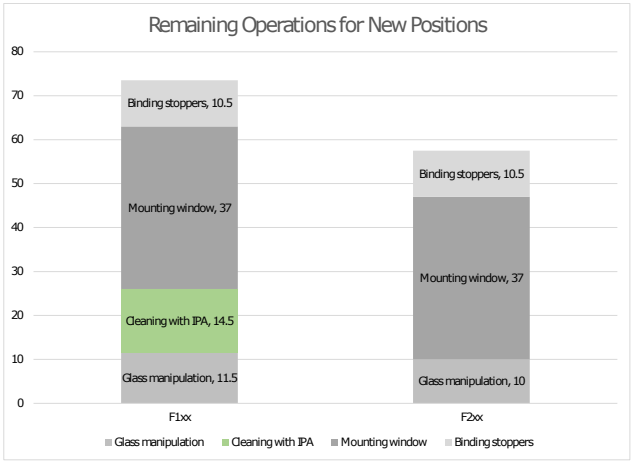
Mounting parts: As discussed previously, mounting parts is a complicated task from a technological perspective. The variety of differing tasks - cable manipulation, screw tightening, snapping sensors into tight cases - would also make automation financially unappealing. The automation must be adapted as TMC develops new car models (or even new sensors). It is hard to predict if the practical life of such an automated robot would be long enough to see a return on initial investments. After a discussion with Mr. Ladislav Kovařík, TMMCZ Supervisor-Specialist, the decision was reached that at least one operator space would be kept on the window preparation line, and its main goal would be mounting parts. Given that in the current state, this task takes 34 seconds for a single pair of windows. The operator still has time to perform more tasks in a total cycle time of 72s.

Binding stoppers: Next, we have decided not to consider automating binding stoppers. These tasks are split into two different versions between our two car models. As such, finding an automated solution for each version would make automation costly since each version takes at most 11s (an average time of 5.5s for both car models per one process). Moreover, stoppers are individualistic for each car model. For future development of new Toyota car models, it can be expected that the current automation of stopper bindings for G1 and G3 would not be usable for new car models (unlike, e.g., general cleaning tasks or gluing dam tasks). Lastly, binding stoppers on G1 is not a manually tricky task that would present a problem from an ergonomics perspective. G3 stopper binding is already automatized thanks to the ultrasonic welding machine the operator can use. The operator, in this case, merely applies a strip of glue (using a brush) on the window and then lets the welding machine finish the operation.

To summarize, as part of our propositions for the automation of the window preparation line, we offer two different options for implementing new operator processes. We call these new positions F1xx and F2xx. F1xx has robotic solutions for gluing dams and applying primer. F2xx automatizes those tasks, too. It also offers a robotic solution for cleaning with IPA, freeing up time for



(a) : Current state of operations.



(b) : New suggested positions.

Figure 4.2: Time comparison of new proposed processes F1xx and F2xx to current situation.

the operator. Time-wise graphs of these options in comparison to the current situation can be seen in Figure 4.2. Possible to automate tasks are shown in green colors; gray tasks are those best left for human operators. Gluing rubber dams is done on average 57s, and primer is applied 33.5s.

Next, we will present our robotic solutions for tasks. All solutions work with the expectation of being applied on a robotic arm (ideally a cobot). Given our largest window dimensions are 1000×1400mm, we have two possibilities for limiting the minimum reach of our robot (as seen in Figure 4.3). Option A has a robot placed above the window preparation line. This allows for robots with a shorter reach of 861mm ($\sqrt{500^2 + (\frac{1400}{2})^2}$). Option B expects the robot to be mounted on a side next to the window preparation line, perpendicular to the window. As such, the robot must have a reach of at least

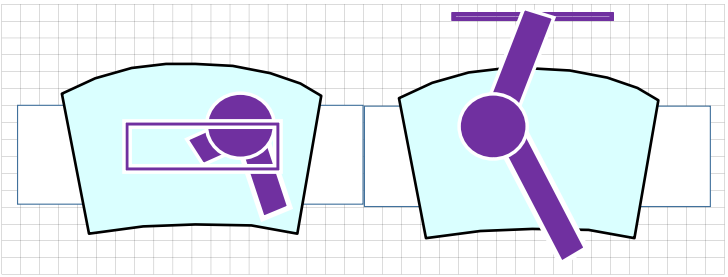


Figure 4.3: Robot placement options.

1221mm ($\sqrt{1000^2 + (\frac{1400}{2})^2}$). The window will be passed to the robot with its wider side parallel to the window line. While this does take more space on the window line (if the window were rotated by 90 degrees, it would save 400mm space on the production line per one robotic workstation), the current window line does have enough space for this type of automation solution. We explain it below.

There are some assumptions we can make about our robotic workplaces. The windows will be securely held by a suction mechanism from below (its type is already used for these processes to help human operators). The window will be positioned predictably and the robot will not need any sensors to determine the state of a window or its position. This will be achieved thanks to two things. First, the robot will always know which window type it is currently working on, thanks to the TMMCZ digital manifest that will be available to the robot's system. Secondly, a mechanical centering system for precisely securing the window position will be implemented. A similar system is already part of urethane robot implementation and can be expected to work similarly for our robotic stations. A rotation drive can be installed on the window line to allow robotic stations to rotate the windows if necessary. Transfer drives will also be added to the line to allow window transfer between workstations (be it robot to robot or robot to human operator).

4.1.1 Robot for Cleaning with IPA

A similar approach to an existing degreasing robot can be applied for this task [5]. A window is loaded on the window preparation line and secured in the robot station using mechanical parts of the line (we can then presume the window is held securely and in the exact position). The robot would perform these actions:

1. Tissues with IPA are cut at the cutting station.

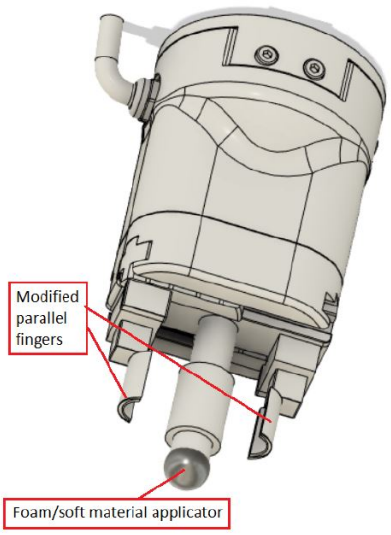


Figure 4.4: Design of EE by Kashyap Suyash [5].

2. Robot with soft grippers picks up prepared tissue.
3. Robot finds the starting trajectory for cleaning and approaches it. The robot recognizes which window is currently on the line from the digital manifest.
4. Robot touches the window and starts cleaning it on given trajectories. Cleaning should be done with ideally roller-wipe motions.
5. Robot finishes the trajectory and moves away from the window.
6. Robot discards used tissue and returns to the cutting station, ready to pick up new tissue for the next window.

The end effector (EE) for this would be a soft-end gripper with two fingers on the side. The fingers approach the soft part of the gripper once the tissue is near and the fingers "hold it in place". Visualization of this can be seen in Figure 4.4.

4.1.2 Robot for Gluing Rubber Dams

Dams are stored wrapped around a cylinder that an outside manufacturer supplies. A robot will be drawing dams from these cylinders. An end effector must divide the dam from its protective tape to reveal the glue. EE must

also cut the dam in desired places using a mechanical slicer. The robot would perform this series of actions:

1. The robot moves over the window to the starting position of gluing the dam to the window.
2. The robot approaches by touching the glass and traverses one dam position's trajectory. The protective tape unwinding mechanism ensures it is removed before gluing dams on the window.
3. After completing one piece of dam trajectory, the slicer in EE is activated, cutting the applied dam piece from the rest of the tape.
4. The robot moves away from the glass.
5. The actions of 2.-4. are repeated for all necessary positions of gluing the dam depending on the window type.
6. After completing all dam gluing trajectories, the robot moves away from the window and returns to the starting position.

A visualization of the possible design of an EE can be seen in a later chapter 6, where we describe our demonstration task.

4.1.3 Robot for Applying Primer

While the operator currently applies primer using a primer bottle with a brush on its end, we suggest a robotic application should instead be based on a tube feeder basis from a larger primer container. A general visualization of such a principle can be seen in Figure 4.5.

We suggest the primer-applying robot perform this series of actions:

1. The robot moves over the window to the starting position of the primer application.
2. The robot approaches by touching the glass. The robot starts applying the primer and moves through the primer application trajectory.
3. After completing the primer application trajectories, the robot moves away from the window and stops applying the primer.

Figure 4.5: Tube feeder general robot.

4. The robot will return to the initial position (the position should be above a point where it would not matter if the primer "drips" by mistake).

4.2 Layout Options

There is a strict order for specific tasks in which they can be performed. These rules are:

- IPA cleaning must come before gluing rubber dams and applying primer. It can, however, happen during or after the mounting of parts.
- Applying primer must happen at the very end, right before the window goes into the urethane robot.

On the other hand, these tasks can happen in any order:

- Mounting parts
- Binding stoppers
- Gluing dams

Given these rules, we can think about robot placement on the window preparation line within a given space. We will now present our four solutions. For each solution, a layout is also shown (one square is 100×100mm). A robot's operating space was estimated to be 2000×1800mm. All financial estimates were created while discussing with Mr. Ladislav Kovařík (Supervisor-Specialist from TMMCZ), Mr. Jaroslav Šmejkal (Omron Technical Sales Representative), and Mr. Kamil Feitl (Omron Programmer). A detailed financial description can be seen in Appendix A.

If robots were to stop working, backup workers would be expected to replace robots in the robot working spaces, while robots would be moved aside and fixed by maintenance workers. It is then essential to keep access to all robots open for temporary backups and maintenance work.

All robots are to be mounted on a side "wall" (i.e., at 90-degree angle), unless stated otherwise. This is a primary choice to avoid a robot frame needed for the "hanging robot". An advantage of a hanging robot is two-sided access. However, the chosen 90-degree robot solution is cheaper. The urethane-robot switchboard will block the primer-robot switchboard from one side in both cases.

4.2.1 Version 1 - Two Robots, no Line Change

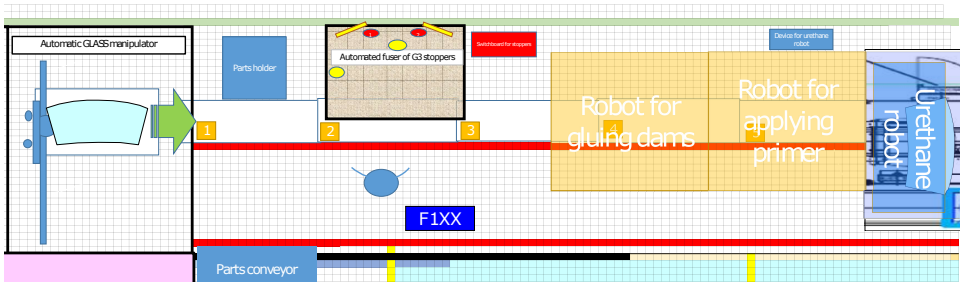


Figure 4.6: Version 1 layout.

The first option we consider uses two robots to glue dams and apply primer. The operator handles mounting parts on the robot, bins stoppers, and cleans the window with IPA (position F1xx). The ultrasonic welding machine for the stopper has to be moved to make space for the dam gluing robot. No other changes to the window line need to be made. The conveyor that supplies parts to the operator remains the same as in the current state. The financial calculation for Version 1 shows that this option could cost around CZK 7 million.

The operator space in Version 1 is the same as the current F108 position. However, both robots require adequate space and zone, which the operator should not enter unless necessary. Enough space must be left between the operator and the robot to keep the operator safe. This proves challenging - gluing the dam robot restricts the space that the operator needs to use to access their working space. In Figure 4.6, red lines denote the operator's working space, and a black line near the bottom of the figure is a safety cage for degreasing-robot on the car rack. As shown in the figure, the space between the degreasing cage and the robot for gluing dams is less than one meter short. This is not ideal from a safety viewpoint - the limited access to the workspace requires the operator to cross between two different robotic applications.

4.2.2 Version 2 - Two Robots, with Line Change

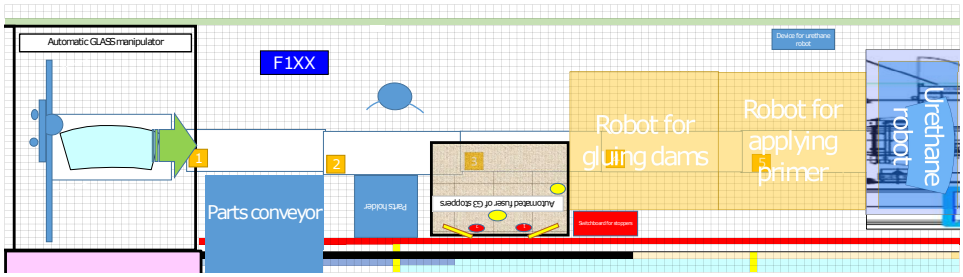


Figure 4.7: Version 2 layout.

The second option also uses two robots (gluing dams and applying primer) while the operator performs F1xx tasks. Version 2 tries to solve the operator's limited access to the workplace by moving the working position to the other side of the window line. In other words, operator in Version 2 has their backs to the trolley row and full access to their working positions (see Figure 4.7).

However, the current space on this side of the window line is only 1000mm wide. This is not enough space to guarantee the operator's safety. To provide more working space (approximately 1600mm as in Version 1), the window line must be moved close to the cage of the degreasing robot. The reconstruction of the window preparation line carries more technological difficulties, as the window line must connect to the automatic window glass manipulator (at the start) and the urethane robot (at the end). As such, both of them would need to be adapted. Moreover, the automatic welder of G3 stoppers must also be moved along with its switchboard to the opposite side of the line.

Parts' conveyors also must change. The current delivery method would be kept, but at its end, a rise conveyor would have to lift the parts up for the

operator to pick them up over the line. All these abovementioned changes have been estimated to cost up to CZK 7.4 million.

4.2.3 Version 3 - Three Robots, Automated IPA Cleaning

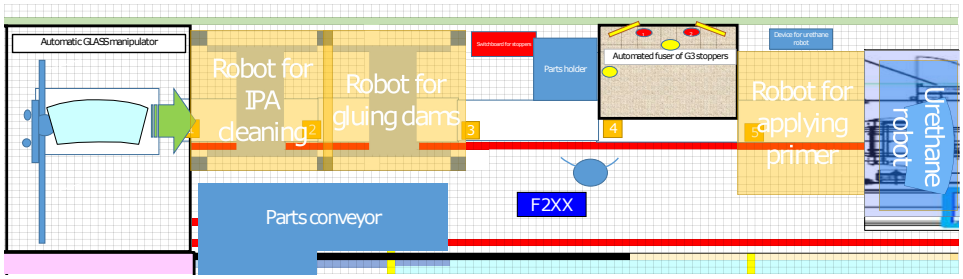


Figure 4.8: Version 3 layout

Version 3 presents three robotic applications on the window line - IPA cleaning, gluing dams, and applying primer. The new operator process would follow option F2xx as described above 4.2. The robots for cleaning IPA and gluing dams are located at the start of the window line, freeing up space between the final primer robot and the degreasing cage. The operator thus has safe access to their workplace. The two robots are also to be mounted "upside down" to both save space on the window line and provide ample space for maintenance to access them.

No hard-line reconstruction would be necessary for this option. Line position, as well as the G3 stopper welder, is kept the same. The parts' conveyor, however, must be adapted; otherwise, access to its current state would be dangerously close to the workspace of robots. A linear conveyor would extend the current situation and deliver boxes to the operator.

Maintenance can access the IPA cleaning robot and dam gluing robots from the trolley row side in case of robot failure. Should one robot fail, only one worker is necessary for its replacement.

The total cost for Version 3 was estimated to be CZK 9 million.

4.2.4 Version 4 - Two Robots, Automated IPA Cleaning

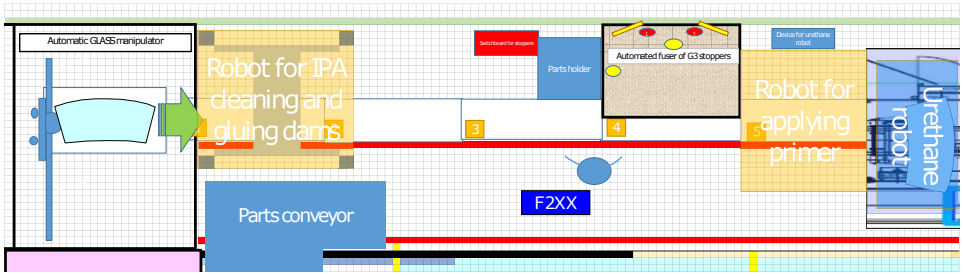


Figure 4.9: Version 4 layout.

Version 4 provides solutions similar to those in Version 3. Three processes are automated (IPA cleaning, dam gluing, and primer application). The operator follows the F2xx process. However, this version uses only two robots instead of using three like in Version 3. A single robot for cleaning with IPA and gluing dam is used here, with a 2-sided gripper with two different end effectors for each task. This makes the robot more technologically challenging to implement but saves on the costs of a third robot that would otherwise be needed. Still, a more technologically demanding gripper can be expected to be more prone to failure than if they were kept "simply" separated (as in Version 3).

In case of failure of the first robot, two backup workers would be required to replace the robot. This is not good, as backup workers are expected to take more time with the tasks due to inexperience. This option is also more expensive than in Version 3 (workers must be paid). Combined with a technologically demanding gripper, it is the most significant disadvantage of this version.

The total expected starting sum was determined to be around CZK 8 million.

4.3 Comparison

The analysis shows that operations F108, F109, and F116 suit automation. The final evaluation of the window line variants can be seen in Figure 4.10.

This evaluation was presented to managers at TMMCZ and discussed with them in depth on December 14, 2023. These were the conclusions from the discussion.

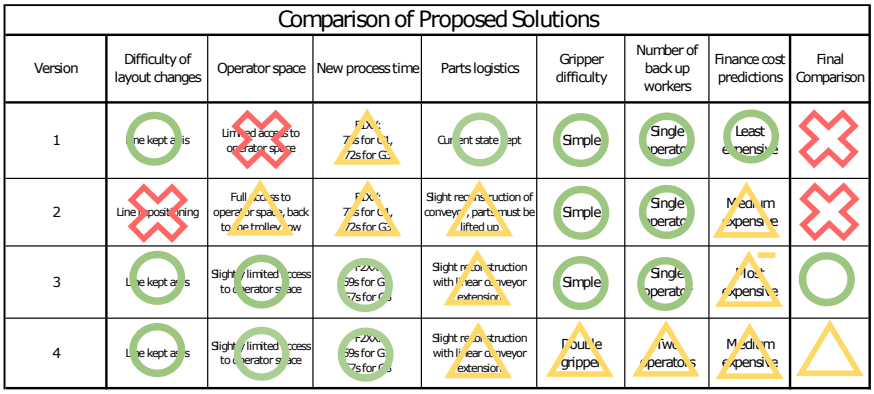


Figure 4.10: Versions comparisons.

The disadvantage of Version 1 (two robots and no line change) is mainly its limited access to the workplace. Otherwise, this version meets the other requirements. The newly proposed operator procedure (F1xx) takes the full factory set cycle time. However, for security reasons, we did not recommend this version.

Version 2 (two robots with line change) requires the most significant intervention in the workspace and the associated rebuilding of the window line. On the other hand, the operator is not limited by space (caused by the cage of robotic workplaces). However, due to the difficulty of rebuilding the window line, we recommended other versions of automation solutions.

Version 3 (three robots with automated IPA cleaning) neither requires a significant rebuild nor dangerously restricts the robot operator's workspace. In this variant, however, three robotic workplaces are expected. Robots are expensive, and the time-saving of the IPA cleaning process is not as significant as other designs. The difference between Version 3 and 4 is CZK 1 million. On the other hand, Version 3 seems safer regarding robot failure and backup required. It can thus be seen as a more viable option in regard to the future outlook of TMMCZ (and not just initial investment).

Version 4 (two robots with automated IPA cleaning) needs a technological implementation of a 2-purpose gripper but solves the problems of previous versions. If the quality and functionality of a multi-purpose gripper can be ensured for these applications, Version 4 would appear to be the best solution. Unfortunately, a single robot failure on the 2-purpose robot would require two operators to back it otherwise up. This makes this option less appealing in regards to the factory's future outlook.

The final choice was to go with Version 3.

Chapter 5

Used Software

Our second assignment in Section 1.2 requires us to demonstrate the task of gluing rubber dams on the rear window of a Toyota Yaris. We make use of a Panda robot from Franka Emika with force/torque sensing capabilities, which is available in the lab of the thesis' supervisor at CIIRC CTU. We seek to control the Panda arm in real time (communication at 1kHz). To program our demonstration task, we used a number of software tools. Our development Github repository can be found at [39].

5.1 Libfranka

Panda robot control is provided over the Franka Control Interface (FCI). FCI provides an interface to control the Panda robot and includes numerous libraries that can provide real-time control. Libfranka is a C++ implementation of FCI's client side. Libfranka takes care of network communication and can provide real-time (and non-real-time) control of the Panda robot, the ability to read sensor data from the robot, and its dynamic model. Real-time communication runs at 1kHz. We can either use libfranka's already implemented motion generators to control the robot over joint position or velocity commands or use a full controller that accepts torque commands.

In our demonstration task, we use force/torque control to command our Panda arm using our computed τ_d in real-time. Libfranka then recomputes

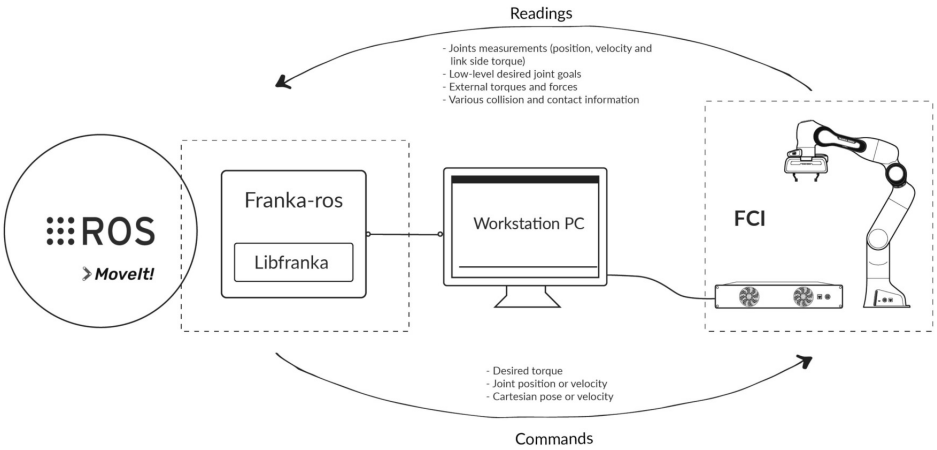


Figure 5.1: Visualization of Panda robot communication with libfranka [2].

out torque command by

$$\tau_c = \tau_d + \tau_f + \tau_g, \tag{5.1}$$

where τ_c is libfranka's real command sent to the Panda robot, τ_f is torque compensation of Panda's motor friction, and τ_g is torque compensation of gravity applied on the robot. Libfranka's model library allows access to forward kinematics (computation of Cartesian pose of last joint from current joint positions), body and zero Jacobian matrices (relates robot's joint movements to its Cartesian velocities), and various dynamics parameters (such as Coriolis and centrifugal vector, inertia and mass matrix and more).

The libfranka library is being developed by Franka Emika company. In April 2023, Franka Emika stopped development on the Panda robot (our robot used for demonstration tasks) and focused on their new Franka Research 3 (FR3) robot arm. Moreover, on September 1st, 2023, Franka Emika filed for insolvency and all development of its products had been frozen. Agile Robots, AG, acquired Franka Emika in November 2023 with hopes of continuing its operations [40]. For these reasons, some of our development on this thesis had to adjust to difficulties regarding outdated libraries and supporting systems.

5.2 ROS2

Robotics Operating System (ROS) is an open-source software library for controlling robots. ROS is often used in education and research robotics development, though commercial use is also possible. ROS currently has two versions - ROS1 and ROS2. ROS1 uses a slave-master interface, while the newer ROS2 uses a Data Distribution Service (DDS). DDS gives better

reliability of communication, is more efficient, has less latency, and provides quality of service checking parameters (QoS). ROS1 does not provide real-time control (though third-party libraries exist), while ROS2 has it implemented in its core. At the time of working on this thesis, ROS2 and Panda robots was only available to program using C++.

For the Panda robot, ROS provides the interface to use libfranka's library within the ROS environment. We can use powerful planners and communication principles of ROS to send commands over libfranka to our robot. For our demonstration task, we have chosen to use ROS2 Humble distribution for its real-time control. Unfortunately, due to Franka Emika's decision to stop support for the Panda robot, ROS2 does not provide access to full implementation to libfranka. Noticeably, under ROS2 it is not possible to access Panda's model library. This blocks us from accessing the robot's current Jacobian and mass matrices and their Coriolis vectors. This information is necessary to compute torque commands for our demonstration task. Third-party libraries (such as Pinocchio library [41]) can help us compute the dynamic parameters of the Panda robot instead.

A work done by our research team at CIIRC used ROS2 and Pinocchio library to control the Panda robot in the spring of 2023. In this project a number of ROS2 real-time controllers were developed, though others could not be implemented due to missing library implementations. Pinocchio library proved to be a powerful tool for computing missing dynamics models under ROS2 implementation. A paper summarizing our achievements can be found in Appendix A named Force_torque_control_of_robot_under_ROS2.pdf.

Moveit2

MoveIt2 is a motion planning platform of ROS2. It provides many options for generating trajectories, helps with planning control navigation, simulating grasping tasks, and can provide 3D visualizations. It utilizes Rviz2 visualization tool. At its base, MoveIt2 is purely a kinematic motion planner, i.e. it plans a series of joint positions for the robot to move through. However, plug-ins exist that can help extend MoveIt2 capabilities to trajectory planning and calculating joint velocities and acceleration paths as well. It is therefore a powerful tool for computing inverse kinematics (joint position computation from EE pose) of robots with custom specified parameters.

5.3 Panda-Python

By the end of September 2023, a new Python language binding library of libfranka was released [42]. These bindings provide full access to libfranka's function implementations, including all its controllers and model libraries. It is easy to use and can be installed within minutes (unlike ROS2 which can take over an hour to initially set up). It comes with libfranka already packed within its installation and works for Panda robot specifically (though the author provides a description of how to use the library for FR3 as well).

During our work on this thesis, we at first started developing our real-time controllers in ROS2 with Pinocchio library. However, after Panda-Python's release we have made the decision to switch to this library instead. Not only do we get immediate access to the robot's actual model dynamics this way without relying on third-party libraries, but we can also make use of powerful Python libraries and toolboxes made for robotics planning and control. Such libraries include Robotics Toolbox (kinematics and dynamics of serial-link manipulators [43]) or Scipy (statistics, optimization, mathematic models [44]).

Chapter 6

Demonstration Task

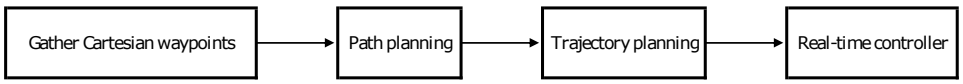


Figure 6.1: Motion planning pipeline.

We present a demonstration task of gluing dams on the rear window of a Toyota Yaris using the torque control of a Panda robot. We implemented a robot motion planning pipeline as seen in Figure 6.1. First, we start by collecting waypoints of our future trajectory. Waypoints are collected in task space (Cartesian space) and, using inverse kinematics, are transformed into joint space. Then, we perform path planning, a purely geometric description of our desired motion in joint space. Using interpolation, we generate a velocity and acceleration profile for our motion, creating a trajectory. Finally, we perform a trajectory control using inverse dynamics and send our desired torques to the robot using our force controller.

6.1 Collecting Waypoints

We must collect a series of poses in task space to generate our path. A pose in three-dimensional space can be defined as

$$W_T = (x, y, z, q_x, q_y, q_z, q_w) \,,$$

where x, y and z are Cartesian coordinates of our pose W_T. The pose rotation is given by quaternion (q_x, q_y, q_z, q_w) in scalar last format.

We have implemented three options for collecting our waypoints.

Using ROS2 and its package rosbag2: We capture an entire trajectory during a hand-guiding mode on the Panda robot. In our code rosbag-extractor.py, we extract joint positions recorded over rosbag, and using roboticstoolbox forward kinematics, we gather all W_T captured over the rosbag2.

Using Panda-Python libfranka bindings: We capture a desired number of poses in the order we wish the path to follow over record_waypoints.py code. This code uses Panda-Python teaching mode to capture joint positions, which we then over roboticstoolbox forward kinematics transform to W_T poses. A smoothing function for captured points was also created in smooth_trajectory.py.

Pregenerate waypoints: We generate our own path using knowledge of the robot environment and window position (file code generate_trajectory.py). We fix Cartesian x-coordinates at 0.55mm in task space. Our y-coordinates are evenly spaced between $[y_{min}, y_{max}]$ values we set at -0.35mm and 0.35mm. For our z-coordinates we use quadratic interpolation to achieve a slight parabolic shape for our window. We define our quaternion as $(1, 0, 0, 0)$, which denotes no rotation for Panda's last joint. We save as poses W_T, ready for our path planner.

All waypoint capture methods mentioned above work within our motion planning pipeline. Our directly captured trajectory over rosbag2 can simulate the direct teaching method (exact replay of hand-guided motions). This can sometimes also copy undesired motions when the human operator's hand shakes or produces uneven paths. Our Panda-Python waypoints method suffers from similar issues. The smoothing function can help with eliminating the worst deviations. Finally, our pregenerated waypoints offer the smoothest movement on window taping demonstration and were used for our final demonstration video capture.

6.2 Planning the Trajectory

With our series of W_T poses saved inside csv files, we are now ready for path planning. We use MoveIt2 and its high-level interface MoveItCpp API. MoveItCpp (unlike its alternative MoveGroup API) is meant for planning for real-time control and industry applications. Our entire code can be seen in

moveit_cpp_traj_gen.cpp. We start by loading in our waypoints W_T and setting up a planning scene for the Panda robot. The interface to the motion planners is through a ROS action or service. There are several motion planners to choose from, the default being OMPL. This is an open-source planning library that mainly implements randomized motion planners. However, we have opted to use the Pilz Industrial Motion Planner instead, a deterministic generator of circular and linear motions. It also supports blending motion segments, which we use in our code. Pilz Industrial Motion Planner is only a motion generator and doesn't provide obstacle avoidance. It does check for the robot's self-collision.

The motion planner must have information about joint limits. By default, the MoveIt2 package named panda_moveit_config has a file joint_limits.yaml with defined limits. It defines the dynamic properties of the Panda robot, precisely its velocity and acceleration limits. Velocity limits are taken from the Franka Emika Panda robot description. Acceleration limits are the highest values that guarantee no jerk limits violation. In the worst case, this is calculated using Euler differentiation from minimum acceleration to its maximum in 1ms.

$$j_{max} = \frac{a_{max} - a_{min}}{0.001s} ,$$

where j_{max} is maximum jerk and a_{max}, a_{min} is the maximum and minimum acceleration possible. Toque limits are not defined by default. However, we have tested implementing true torque limits with actual true acceleration limits inside our joint_limits.yaml. The resulting trajectory produced by the MoveIt2 motion planner proved to be highly unstable for the actual robot controller in such a case, resulting in high-speed motions that caused the Panda robot to enter error states. For this reason, we decided to keep the default settings of joint_limits.yaml and use them for our actual force controller.

We had to define motion planner parameters to set up our planning motion request. These can be seen in file moveit_cpp_traj_gen.yaml. Pilz Industrial Motion Planner offers several motion command alternatives. We have implemented two of them and can easily switch between them by editing the configuration file. The first is "PTP" planner - a synchronized point-to-point trajectory planner. The second planner is the "LIN" motion command. This generates a linear Cartesian trajectory between the goal and start pose. All motions are synchronized - translation is calculated over linear interpolation, and rotation motion is a spherical linear interpolation.

Both planners accept Cartesian poses as their input for planning and return joint trajectory. MoveIt2 is a kinematic motion planner - by default, it does not plan time-wise velocity and acceleration profiles. We plan a motion path between two consecutive waypoints using a Pilz planner and save their result.

After we plan for all waypoints this way, we can call for different MoveIt2 trajectory post-processing plug-ins. We first call a Time-Optimal Trajectory Generation and use it to produce smooth and continuous velocity profiles for our entire path (not just between two consecutive waypoints, but for their whole series). The method is described in paper [45]. This method can introduce a circular blend (deviation) around waypoints. Since our desired final trajectory should follow a straight line, this possible blend only helps smooth out our path. A second MoveIt2 plugin we use is a Ruckig smoothing algorithm, which produces a jerk-limited trajectory [46]. To use it, we had to define jerk limits in our joint_limits.yaml file.

The MoveIt2's result velocity and acceleration trajectory profiles can be used in real-time torque controllers using a number of possible interpolations. We have tested Hermite's interpolation, which used MoveIt2's planned trajectory (position, velocity and acceleration profiles) to control the Panda robot. However, the results were highly unstable, often forcing the Panda robot into error states for violating force limits. As such, we decided to use cubic spline interpolation instead, only using MoveIt2's planned path and position profile (described in more detail below).

We save our result joint trajectory into a csv file. Now, we are ready to use our force controller.

6.3 Realtime Controller

We have implemented our real-time controller over the Panda-Python bindings of libfranka. While real-time control is also possible in ROS2, due to the development of Panda robot having been discontinued, its current ROS2 libfranka binding is not fully implemented and requires third-party libraries to access information about Jacobian, Coriolis forces and more (for example, we have tested the pinocchio library and found it to be a fairly accurate replacement). For this reason, we have chosen to use Python binding of full libranka to get access to all its functions.

Our code implementation is in force_controller_cubic.py file. Before we can use it, we must connect to the Panda robot. This can be done manually over a terminal or use our code log_in.py, which unlocks the robot and logs the user's computer into Panda Desk (FCI client-side) as the current user. Our controller uses a torque command interface, so we must implement a way to transform our joint positions to torque commands. We do this by using interpolation and then performing inverse dynamics.

	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7
Velocity limits $[\frac{rad}{s}]$	2.0	2.0	2.0	2.0	2.35	2.35	2.35
Acceleration limits $[\frac{rad}{s^2}]$	3.75	1.875	2.5	3.125	3.75	5.0	5.0

Table 6.1: Dynamic limits of the Panda robot for our force controller

6.3.1 Interpolation

From MoveIt2 path planning, we get a series of waypoints in joint positions. We plan on using a real-time controller that needs to know the exact desired joint positions, velocities, and acceleration at any given time. Interpolation is an estimation method used in robotics for constructing new continuous joint positions in time out of a series of known joint positions in discrete times.

First, we find discrete times for our joint positions. We could use timestamps provided by MoveIt2 Time Optimization Planner, but we opted to use and calculate our own timestamps. This lets us modify already planned paths more easily and depending on our needs. We defined our velocity and acceleration limits as seen in Table 6.1. These limits were derived as approximately 90% maximum allowed velocity limit and acceleration limits as Euler differentiation in a worst-case jerk as described previously.

We calculate our discrete timestamps from the maximum needed time for lead axis movement (an axis with the longest joint position movement) while honoring velocity and acceleration limits. The time equations for single joint movement were

$$t_{velocity} = \frac{q_i - q_f}{v_{limits}} ,$$

$$t_{acceleration} = \sqrt{\frac{q_i - q_f}{a_{limits}}} ,$$

where $t_{velocity}$ and $t_{acceleration}$ are times honoring velocity and acceleration limits, q_i and q_f are the initial and final joint position and v_{limits} and a_{limits} are velocity and acceleration limits as seen in Table 6.1. Our final time t for motion between q_i and q_f for the single joint is chosen as $t = \max(t_{velocity}, t_{acceleration})$. We perform this calculation for all seven joints and choose the maximum required t.

With our joint positions marked with timestamps generated by us, we can

set up their interpolations. We have chosen to use cubic spline interpolation from SciPy interpolate library. The cubic spline is a piece-wise linear interpolator that is continuous even in its derivatives. If we suppose our time t is always rising (even if not equally spaced) and data $y_0, ..., y_n$ given, we find cubic spline polynomial $S_k(x)$ as

$$S_k(x) = a_k + b_k \cdot (x - t_{k-1}) + c_k \cdot (x - t_{k-1})^2 + d_k \cdot (x - t_{k-1})^3 \,,$$

where a_k, b_k, c_k, d_k are values to be determined and k denotes spline on interval $[t_{k-1}, t_k]$. We implement constraints to ensure cubic spline smoothness, such as continuity in its function values and both derivatives and natural boundary conditions (the second derivative is always zero at endpoints). The Cubic Spline method from scipy.interpolate also offers easy computation of its first and second derivatives (i.e., our desired velocity and acceleration). Graphs for our cubic spline interpolation for the final demonstration task can be seen in Figure 6.2.

Thanks to interpolating our pre-calculated MoveIt2 joint position values, we can find desired joint positions, velocities, and accelerations at arbitrary time of our controller.

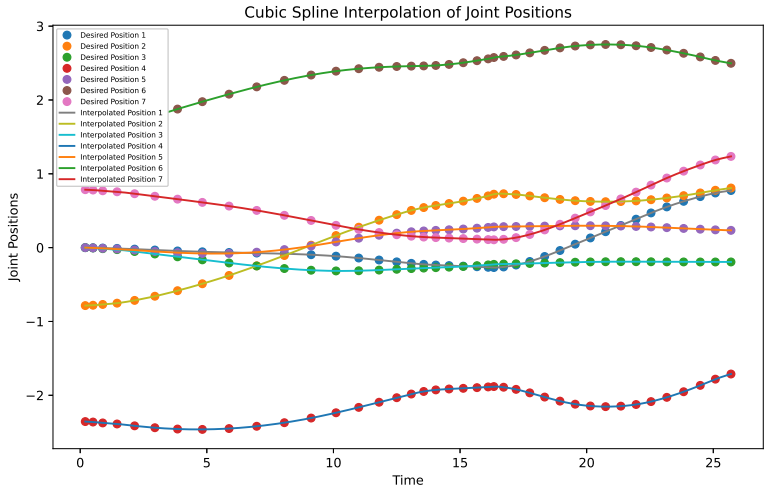
6.3.2 Inverse Dynamics

Inverse dynamics provide joint torques based on joint positions, velocities, and acceleration. We use Lagrangian dynamics, defined as

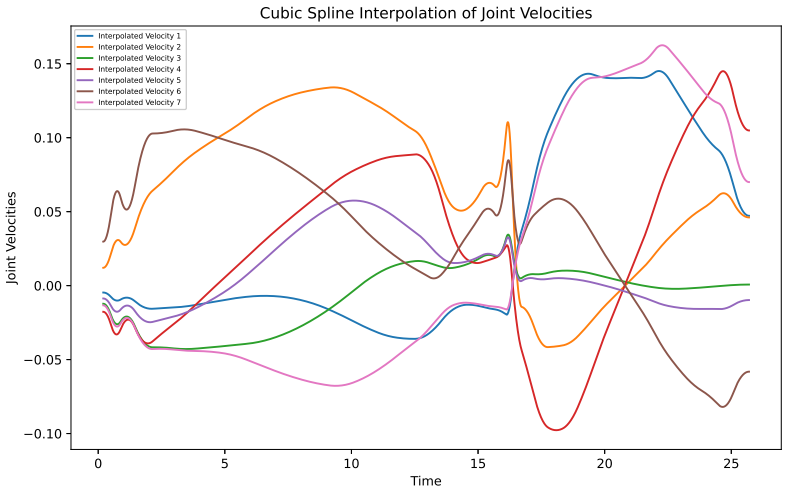
$$\tau_L = M(q) \cdot \ddot{q} + C(q, \dot{q}) \cdot \dot{q} + g(q) \,,$$

where $M(q)$ is mass matrix of system joint configuration q, $C(q, \dot{q})$ is Coriolis and centripedal terms for given joint configuration and $g(q)$ is gravity term. The q, $\dot{q}$, $\ddot{q}$ are current joint positions, velocities, and accelerations. Thanks to libfranka's implementation (as shown in equation 5.1), we do not have to compensate for the gravity term. Thus, we only need to compute

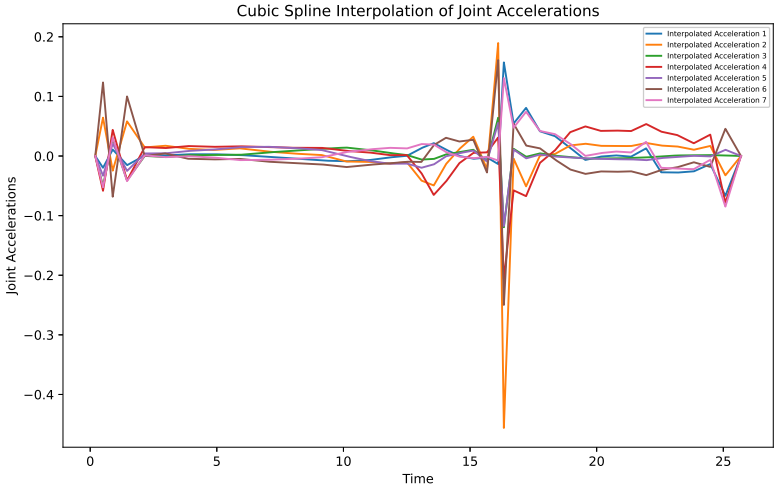
$$\tau = M(q) \cdot \ddot{q} + C(q, \dot{q}) \cdot \dot{q}$$



(a) : Position interpolation



(b) : Velocity interpolation.



(c) : Acceleration interpolation.

Figure 6.2: Cubic spline interpolation for our final demonstration task.

Using Panda-Python bindings of libfranka, we can extract the coriolis and mass matrix of the system at any desired time and joint configuration. We also introduce a feedback control $\tau_{feedback}$ term as PD control of our current position and velocity. The equations are given by

$$e = q_m - q_d \,,$$

$$\dot{e} = \dot{q}_m - \dot{q}_d \,,$$

$$\tau_{feedback} = k_P \cdot e + k_D \cdot \dot{e} \,,$$

where q_m, $\dot{q}_m$ are measured joint positions and velocities, q_d, $\dot{q}_d$ are desired joint positions and velocities and k_P, k_D are constant gains. Gains were determined experimentally based on system behavior, and their final values for our controller can be seen in table 6.2.

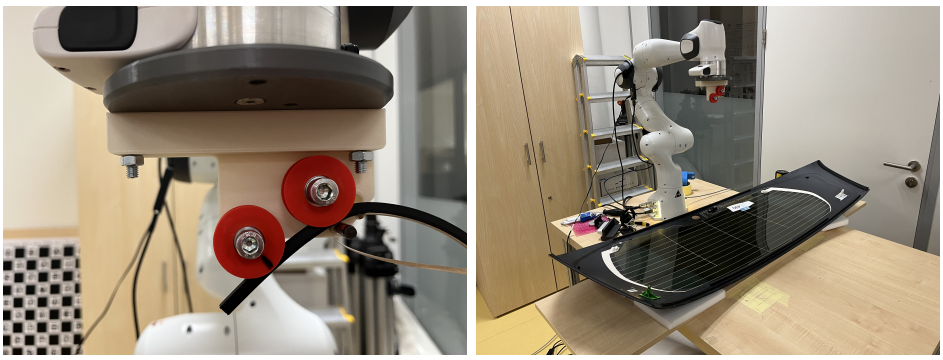
k_P	100	120	100	100	50	20	17
k_D	15	17	15	10	6	6	4

Table 6.2: Force controller PD gain constants

Our final τ_d (torque desired) was calculated as

$$\tau_d = \tau - \tau_{feedback}$$

6.4 Demonstration Gripper



(a) : Our designed end effector. **(b) :** Workspace.

Figure 6.3: Demonstration task environment setup.

We have designed an end effector that glues the dam on the window. Its design files can be found in Appendix A. We used the Onshape Free CAD platform to design EE, create its STL files, and then print it on Prusa 3D printer in our CIIRC laboratory. The main EE body has two wheels on bearings attached to it. The first wheel helps position rubber dams on the EE and guides it into line with the second wheel. As the dam passes the

first wheel, the protective tape is stripped free from the dam, and a metal dowel pin helps with unwinding the protective tape. Then, the second wheel positions the dam on the window and, with force applied perpendicularly to it over gravity, glue the dam onto the window.

The front view sketch of our gripper can be seen in figure 6.4. The shortest rubber dam glued on the robot is 70mm (G3, RR). Our EE must be able to glue a dam of such size. Rubber dams are stiff enough that by unwinding the protective tape from the rubber dam after it passes the first wheel, the resulting force "pushes" the rest of the dam into our second wheel, which then glues it on the window. The distance between our wheels cannot be larger than 70mm to ensure even the smallest dam can be glued on the window using our method. In our final design, the distance between the center of our wheels was set at 41.2mm, or in other words, the distance dam had to pass between our two wheels was 30mm (as the remaining 11.2mm was the outer diameter of the extrude that held our ball bearings in place on our EE).

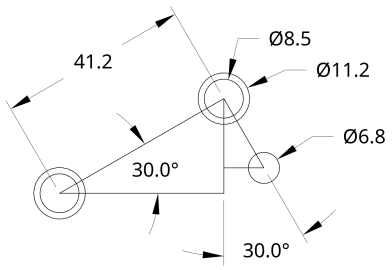


Figure 6.4: Sketch of EE main face.

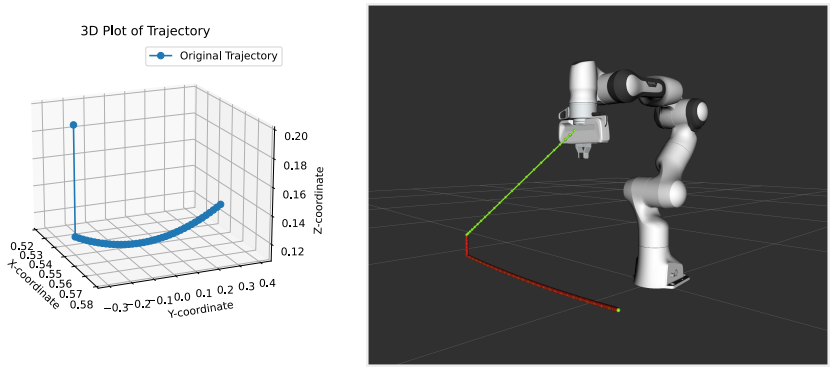
The wheels were designed to hold our dams securely, and the wheels' inner shape copies rubber dams exactly. We used ball bearings 608 for our wheels and designed our plastic wheels to have a radius of 17mm. This way, the 5mm outer ring of our wheels was just enough to hold our dams and allow us to passively slide the wheel on the window. Our EE places our main gluing wheel on the rotation axis of the last Panda robot joint and its flange. This helps simplify motion planning for our movements even when the EE must "rotate" around corners (e.g., G3's FR) and avoids unnecessary robot movements that could otherwise be caused by the robot trying to rotate around an unaligned axis.

The design created as part of this thesis work is purely passive, but two possible mechanical enhancements were considered during its creation. First, a mechanical slicer will be placed between our two wheels to cut rubber dams to our desired size. Second, an automatic winder of protective tape after its removal from the rubber dam. In our thesis work and its demonstration task, we instead pre-cut dams into desired sizes and used human work to separate wind-protective tape from rubber dams. This option was chosen due to time

constraints.

6.5 Task Demonstration

We chose a pipeline for our final task demonstration of gluing dams on the window: pre-generated waypoints, a "PTP" MoveIt2 path planner, and a force controller over panda-python libfranka bindings. Our generated path can be seen in figure 6.5. Our first waypoint is slightly above our starting position on the window to ensure our gripper lands on the window correctly and starts taping the dam in our desired position.



(a) : Plot of the generated path. **(b) :** MoveIt2 planned path.

Figure 6.5: Generated path for dam gluing demonstration task.

Appendix A shows a video of our task demonstration. In Figure 6.6, our recorded joint position over controller time can be compared to our desired joint positions reached simultaneously. Our controller deviation error from desired positions was less than 1% maximum for all joints (exact values can be seen in table 6.3, calculated by taking the entire range of joint limits and maximum deviation $error = \frac{maximum\ deviation}{total\ joint\ range} \cdot 100$).

	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7
Error [%]	0.47	0.62	0.28	0.99	0.66	0.90	0.78

Table 6.3: Controller deviation error from desired positions

Despite our reasonably low error, we have observed a strange deviation from the planned trajectory after it passes through the middle trajectory point in the y-axis (around 4mm, can be seen in figure 6.7). We speculate the reason for this deviation is our Panda arm passing by a singularity - a point in the robot's workspace where the robot loses degrees of freedom and starts behaving unexpectedly. Singularities are also often accompanied by sudden acceleration changes. We can see in Figure 6.2 that our planned

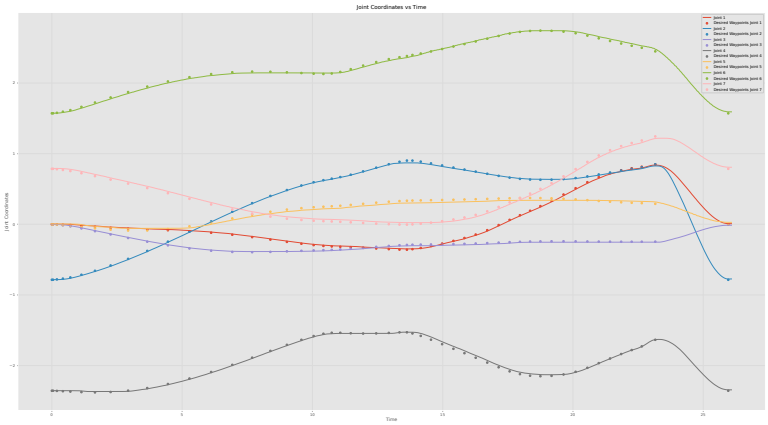


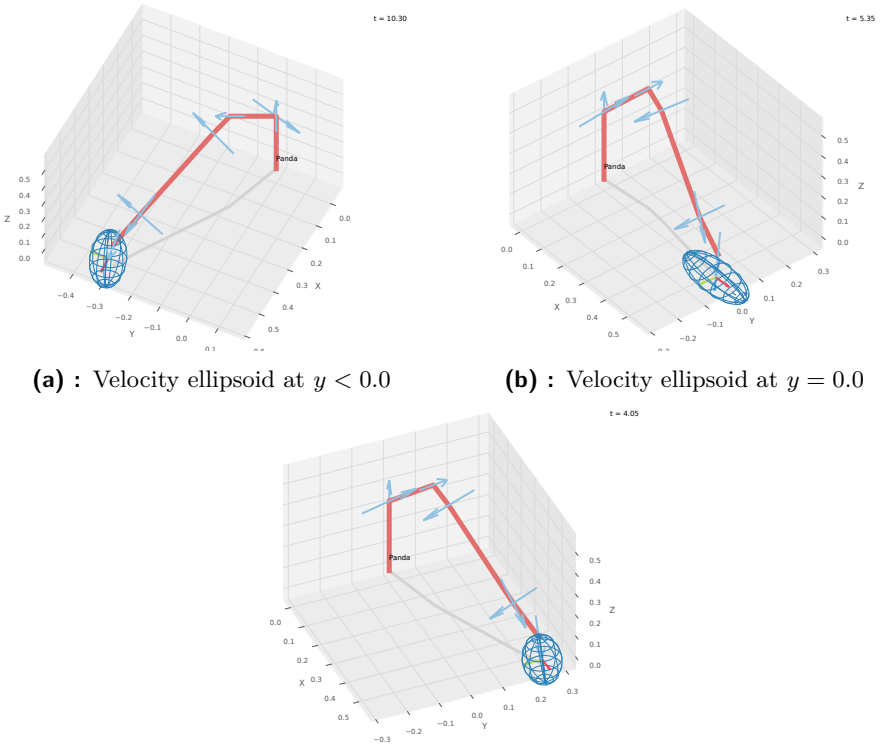
Figure 6.6: Graph of real robot joint positions in time compared with our desired positions.

interpolation acceleration at around $t = 16s$ suddenly accelerates in all its joints. In the same moment, our velocity interpolation graph shows a sudden "switch point", where all velocities change signs. We decided to investigate this point further by looking at the velocity manipulability of the ellipsoid as it passes the midpoint in the y-axis. Velocity ellipsoid helps us visualize the robot's arm's ability to perform motion along its axes - the larger the axis, the easier the movement. We discovered that when the robot arm passes $y = 0.0$, the lead axis of the velocity ellipsoid changes orientation from one side to the next (figure 6.8), meaning that in our midpoint in the y-axis, the robotic arm has the lowest manipulability in y-axis at that point.

Figure 6.7: Demonstration task path error. The red line denotes the desired path and our deviation from it.

In an industrial setting, this deviation would, of course, not be allowed. However, our situation in a laboratory at CIIRC is different from what TMMCZ would have access to. Our Panda robot has a reach of 850mm. This barely covers the smallest window (G1's RR). Our robot is also mounted on the same desk as the window is placed on. This will most likely not

be the case in TMMCZ; as per our suggested options for automation tasks, robotic arms should be placed on a side wall (perpendicular to the window placement) or above the window. TMMCZ will also most likely use a robotic arm with greater reach (e.g., Omron's TM12 with 1300mm reach) to avoid any possible issues with forcing the robotic arm into joint limits. As such, we have determined our issue with path deviation to be allowable for our laboratory demonstration task testing.



(a) : Velocity ellipsoid at $y < 0.0$

(b) : Velocity ellipsoid at $y = 0.0$

(c) : Velocity ellipsoid at $y > 0.0$

Figure 6.8: Velocity ellipsoid transformation.

Chapter 7

Conclusion and Future Work

The two assignments for the diploma project given in Section 1.2, named Part 1 and Part 2 were fulfilled, in our humble opinion.

Part 1 studied how to robotize operations on a window line in the final assembly shop of TMMCZ. We analyzed three processes on the window preparation line. The Toyota-related experience and practices were mediated by the supervisor-specialist Mr. Ladislav Kovařík and his team. The automation options for five tasks were studied. These tasks include cleaning windows with IPA, mounting parts on windows like the rear mirror, binding stoppers, gluing rubber dams, and applying primer.

Two tasks are not suitable for automation. Mounting parts presents a technologically complex robotics problem. Binding stoppers is not a generic enough task to be viable from a financial perspective. It also does not fit the TMMCZ outlook. The three remaining tasks were analyzed in detail. Robotic solutions were suggested and studied for IPA cleaning, dam gluing, and primer application.

We have presented four variants of how to robotize the window preparation line. The analysis dealt also with possible production line layout modifications inducing modified robotic workspaces. Technological, financial, and economic aspects were analyzed as well. The four variants are (Variant 1) two robots with no line change; (Variant 2) two robots with a line change ; (Variant 3) three robots with automated IPA cleaning; and (Variant 4) two robots with automated IPA cleaning. These variants were presented at a meeting with TMMCZ higher management at the beginning of December

2023. The managers considered our proposed variants promising for the future of TMMCZ and other Toyota manufacturing plants. The critical discussion at the meeting resulted in the recommendation of TMMCZ managers to work on Variant 3 in the foreseeable future.

Variant 3 (Three robots, automated IP cleaning) can be implemented either using ordinary industrial robots or force/torque-compliant robots. The former option has been currently followed in TMMCZ in other robotized cells. The latter option using force/torque-compliant robots, has not been used in TMMCZ yet. The higher safety makes this solution attractive.

This situation was foreseen in the diploma project assignment. It led to Part 2 (Demonstrating the use of a force/torque-compliant robot) of the thesis, which was designed and implemented in the CIIRC CTU laboratory.

We have developed a motion planning pipeline for the Panda robot to implement the demonstration task – gluing the dam on a car window. We collected a series of waypoints we needed to pass through. It was done manually using a robot teach pendant. The outcome is the geometry of the required movement. A trajectory planning tool MoveIt2 provides a trajectory in the configuration space, i.e., separately for each robot joint. The resulting trajectory adds velocities and accelerations to the known motion geometry. The recently released Python bindings (Panda-Python) provided real-time force/torque control to the Panda robot. The described procedure and its implementation constitute a torque control. The expected demonstration was implemented and practically tested. We also compute robot trajectory from our planned path by implementing our timing rule to define desired joint velocities and acceleration. We use cubic spline interpolation to generate the desired trajectory's position, velocity, and acceleration profiles. Finally, we perform inverse dynamics to calculate torque command for the robot from desired joint positions, velocities, and accelerations. Our torque controller has also implemented feedback control to ensure the correct path is followed.

We designed a passive mechanical tape gluing end effector that performs the dam gluing. This end effector was designed, its blueprint created in CAD software, printed on a 3D printer and assembled in the CIIRC CTU laboratory. The design considers the stiffness of rubber dams when implementing their supply into the end effector. The end effector was designed with the current implementation of the gluing dams task at TMMCZ in mind, including the dam sizes and path on all four window types.

A minor deviation from the expected trajectory was noticed when experimenting with gluing the dam demonstration task. We documented this error

and explored its origins. The Panda robot approached a singularity, and its velocity ellipsoid was considered. This error would be avoided in the industrial setting using robotic arms better fitted for Toyota Yaris and Aygo X window sizes.

Our entire software development process is available in CIIRC CTU Gitlab repository [39].

There are several possible directions to continue the presented work. First, it is recommended to continue working with TMMCZ specialists and managers to elaborate Version 3 into a fully detailed specification of window line preparation. It could serve as the assignment to an integrator, a technological company, which would implement the solution at TMMCZ. Second, it is possible to endow the end effector with active force/torque elements. This active feedback in the grippen might allow to use of ordinary robots. It would be possible to implement the suggested improvements on the dam gluing end effector.

Appendix A

Attached Files

A list of all attached files is in Table A.1.

File name	File description
EEMainDesign.pdf	Design of end effector main body
EEWheelDesign.pdf	Design of end effector wheels
FinancialCalculation.pdf	Detailed financial calculations
ForceTorqueControlOfRobotUnderROS2.pdf	Research project of spring 2023
VajnerovaLucieDiplDemo1920x1080.mp4	Demonstration task video recording

Table A.1: Attached files

Appendix B

Bibliography

[1] Chang InSung, Cho YongJoon, Park HyunSung, and So DeugYoung. Importance of fundamental manufacturing technology in the automotive industry and the state of the art welding and joining technology. *J Weld Join*, 34(1):21–25, 2016.

[2] Franka Emika. *Panda Arm*, 2020. Product Manual.

[3] Eric Rosales, Qiang Gan, and John Gan. Forward and inverse kinematics models for a 5-dof pioneer 2 robot arm. *Technical Report*, 01 2002.

[4] Lynn McAtamney and E. Nigel Corlett. Rula: a survey method for the investigation of work-related upper limb disorders. *Applied Ergonomics*, 24(2):91–99, 1993.

[5] Kashyap Suyash. *Automated degreasing application for car window body flange*. Bachelor's thesis, Czech Technical University in Prague, Prague, Czech Republic, January 2022. Available at `https://dspace.cvut.cz/handle/10467/102706`.

[6] Amith Kulkarni, P Dhanush, B S Chetan, C S Thamme Gowda, and Shrivastava Prashant Kumar. Recent Development of Automation in Vehicle Manufacturing Industries. *International Journal of Innovative Technology and Exploring Engineering*, 8(6S4):410–413, jul 26 2019.

[7] Orlando Melendez, Ravindra Thamma, and Daniel Kirby. Automation in the automotive industry. *International Research Journal of Modernization in Engineering Technology and Science*, 02(10):282–286, 2020.

[8] Setia Hermawati, Glyn Lawson, Mirabelle D'Cruz, Frank Arlt, Judith Apold, Lina Andersson, Maria Gink Lövgren, and Lennart Malmsköld.

Understanding the complex needs of automotive training at final assembly lines. *Applied Ergonomics*, 46:144–157, 2015.

[9] Charles Wankel. *Toyota Motor Corporation*, chapter 2, pages 1586–1588. SAGE Publications, Inc., 2455 Teller Road, Thousand Oaks California 91320 United States, 2009.

[10] Thembani Nkomo. Strategy - analysis of toyota motor corporation, 2012.

[11] Toyota Production System. `https://www.toyotacz.com/en/about-us/toyota-world/toyota-production-system/`.

[12] Toyota Kolín. `https://www.toyotacz.com/en/about-us/toyota-kolin/`.

[13] andrewbiddle. Toyota Assumes Full Ownership of the Kolin Car Plant in the Czech Republic. `https://media.toyota.co.uk/toyota-assumes-full-ownership-of-the-kolin-car-plant-in-the-czech-republic/` jan 1 2021.

[14] Isak Karabegović, Edina Karabegović, Mehmed Mahmic, and Ermin Husak. *The Implementation of Industry 4.0 by Using Industrial and Service Robots in the Production Processes*, pages 656–685. IGI Global, 2021.

[15] Zuzana Papulová, Andrea Gažová, and Ľubomír Šufliarský. Implementation of Automation Technologies of Industry 4.0 in Automotive Manufacturing Companies. *Procedia Computer Science*, 200:1488–1497, 2022.

[16] Wanek Golnazarian and Ernest Hall. Intelligent industrial robots. In *IMTS 2002 Manufacturing Conference*, 09 2002.

[17] Dr Ramanajneyulu and PROF MOHAN. Logistics concept of supply chain in automotive production. *International Journal of Asian Management*, Volume 1, I:pp. 85–90, 07 2023.

[18] Nils Boysen, Simon Emde, Michael Hoeck, and Markus Kauderer. Part logistics in the automotive industry: Decision problems, literature review and research agenda. *European Journal of Operational Research*, 242(1):107–120, 4 2015.

[19] Simon Emde and Nils Boysen. Optimally locating in-house logistics areas to facilitate jit-supply of mixed-model assembly lines. *International Journal of Production Economics*, 135:393–402, 01 2012.

[20] Jorge Carrillo, Adriana Martínez, Octavio López, and Martha Díaz. The automotive sector in Mexico: The impact of automation and digitalization on employment, apr 18 2023.

[21] Philippe Aghion, Celine Antonin, Simon Bunel, and Xavier Jaravel. Modern manufacturing capital, labor demand and product market dynamics: evidence from France. CEP Discussion Papers dp1910, Centre for Economic Performance, LSE, March 2023.

[22] Benjamin Moll, Lukasz Rachel, and Pascual Restrepo. Uneven growth: Automation's impact on income and wealth inequality. *Econometrica*, 90(6):2645–2683, 2022.

[23] Uqba Othman and Erfu Yang. Human–robot collaborations in smart manufacturing environments: Review and outlook. *Sensors*, 23:5663, 06 2023.

[24] Lihui Wang, Xi Vincent Wang, József Váncza, and Zsolt Kemény. *Advanced Human-Robot Collaboration in Manufacturing*. Springer Nature, jun 10 2021.

[25] Martin J. Rosenstrauch and Jörg Krüger. Safe human-robot-collaboration-introduction and experiment using iso/ts 15066. In *2017 3rd International Conference on Control, Automation and Robotics (ICCAR)*, pages 740–744, 2017.

[26] Sek Kian, Eugene Tee, Hashim Low, Wan Saim, Wan Nurshazwani, Safinaz Zakaria, Mohd Binti, Hazlita Khialdin, M Isa, Chin Awad, Kian Tee, Low Eugene, I.Saim Hashim, W.N Wan Zakaria, Safinaz Binti, Safinaz Mohd Khialdin, Hazlita Isa, Mohammed Awad, and Chin Soon. A study on the ergonomic assessment in the workplace. In *Advances in Electrical and Electronic Engineering: From Theory to Applications: Proceedings of the International Conference on Electrical and Electronic Engineering*, volume 1883, 08 2017.

[27] Edward Fu, Karen McCue, and Diane Boesenberg. F.1 - chemical disinfection of hard surfaces – household, industrial and institutional settings. In Ingegärd Johansson and P. Somasundaran, editors, *Handbook for Cleaning/Decontamination of Surfaces*, pages 573–592. Elsevier Science B.V., Amsterdam, 2007.

[28] Zhenjing Li and Lap Tam. A survey on techniques and applications of window-cleaning robots. *IEEE Access*, PP:1–1, 08 2021.

[29] Qiang Zhou and Xin Li. Experimental comparison of drag-wiper and roller-wiper glass-cleaning robots. *Industrial Robot: An International Journal*, 43:409–420, 06 2016.

[30] Jihong Zhu, Andrea Cherubini, Claire Dune, David Navarro-Alarcon, Farshid Alambeigi, Dmitry Berenson, Fanny Ficuciello, Kensuke Harada, Jens Kober, Xiang Li, Jia Pan, Wenzhen Yuan, and Michael Gienger. Challenges and outlook in robotic manipulation of deformable objects. *IEEE Robotics and Automation Magazine*, 29(3):67–77, 2022.

[31] Siyu Lin, Xin Jiang, and Yunhui Liu. Cable manipulation with partially occluded vision feedback. In *2022 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1245–1250, 2022.

[32] Yu She, Shaoxiong Wang, Siyuan Dong, Neha Sunil, Alberto Rodriguez, and Edward Adelson. Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research*, 40(12-14):1385–1401, 2021.

[33] Ling Tang and Yan-Bin Jia. Robotic fastening with a manual screwdriver. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5269–5275, 2023.

[34] Shivraman Thapliyal. Ultrasonic welding—a modern welding technology for metals and plastics. In J. Paulo Davim, Kapil Gupta, Kapil Gupta, and J. Paulo Davim, editors, *Advanced Welding and Deforming*, Handbooks in Advanced Manufacturing, pages 1–22. Elsevier, 2021.

[35] Yasuhiro Kuroda and Hideo Suzuki. Dam rubber for car window, aug 11 1990. Patent no. JPH02274615A.

[36] Qilong Yuan, I-Ming Chen, Teguh Lembono, Simon Landén, and Victor Malmgren. Strategy for robot motion and path planning in robot taping. *Frontiers of Mechanical Engineering*, 11, 05 2016.

[37] Justin C. Bolger. *Structural Adhesives: Today's State of the Art*, pages 133–194. Routledge, oct 24 2018.

[38] Teroson. *Polyurethane Black Primer*, 10 2017. Product Datasheet.

[39] Lucie Vajnerová. Lucie Vajnerová / Masters Devel · GitLab. `https://gitlab.ciirc.cvut.cz/vajneluc/masters-devel`, dec 18 2023.

[40] Agile Robots. Agile Robots AG acquires robotics specialist Franka Emika. `https://www.agile-robots.com/en/news-events/detail/agile-robots-ag-acquires-robotics-specialist-franka-emika`, nov 2 2023.

[41] Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard. The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *IEEE International Symposium on System Integrations (SII)*, 2019.

[42] Jean Elsner. Taming the panda with python: A powerful duo for seamless robotics programming and integration. *SoftwareX*, 24:101532, 2023.

[43] Peter Corke and Jesse Haviland. Not your grandmother's toolbox–the robotics toolbox reinvented for python. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11357–11363. IEEE, 2021.

[44] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[45] Tobias Kunz and Mike Stilman. Time-optimal trajectory generation for path following with bounded acceleration and velocity. In *Robotics: Science and Systems*, pages 09–13, July 2012.

[46] Lars Berscheid and Torsten Kröger. Jerk-limited real-time trajectory generation with arbitrary target states. *Robotics: Science and Systems XVII*, 2021.