

Automated Defects and Failure Inspection in Remanufacturing and Maintenance

Zezhong Wang

Doctor of Philosophy

Aston University

March 2024

©Zezhong Wang, 2024

Zezhong Wang asserts their moral right to be identified as the author of this thesis

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright belongs to its author and that no quotation from the thesis and no information derived from it may be published without appropriate permission or acknowledgement.

Abstract

The quality of the key components of manufacturing equipment, such as dies and moulds in the manufacturing industry, is highly relative to the quality of final products. Since dies/moulds work in harsh situations, subsurface defects happen and will gradually develop into fatal surface cracks. Therefore, regular maintenance and remanufacturing are necessary for dies/moulds. Inspection is a key process in implementing maintenance and remanufacturing that can detect and prevent the components from fatal damage. As an important non-destructive testing (NDT) inspection method, ultrasonic testing (UT) can detect subsurface cracks. Currently, most industrial UTs are still carried out manually. In this study, an automated robotic UT on dies/moulds is studied to improve the efficiency and effectiveness of inspection in remanufacturing/maintenance.

In contact industrial UT, the probe must contact the object surface and be normal to the surface to guarantee the amplitude of the received ultrasonic waves, optimising the inspection result. The moving speed on locations, such as edges, should also be considered to maintain smooth scanning. Therefore, the results of UT heavily depend on the expertise level of operators.

In this study, the objective is to implement UT with a UT probe attached to the end-effector of a robotic arm. The object is placed at the assigned location on a desk surface. The robotic arm will approach the assigned waypoint to scan the complex surface of the object without computer vision. A reinforcement learning (RL) model is introduced to control the orientation and moving speed of the UT probe during UT scanning. Considering the safety reason and precision of operation, a collaborative robotic arm, UR5e, is used to carry out UT. Only a 6 DOF force/torque sensor measures contact force/torque between the end-effector and the surface. The measured forces are used as interactions between the robot and the environment to give feedback to the RL model so that it can make action decisions to adjust the orientation and the moving speed of the UT probe. At each waypoint, the end effector's orientation adjustment and the moving speed will be planned in real-time by the “brain”, i.e., an A2C RL model. As the “muscle”, a model-based compliance controller will be used to maintain the contact force between the end-effector of the robotic arm and the object surface constant to keep the probe contacting the object surface. A control software platform based on the robot operation system (ROS) is established to implement the whole methodology in simulation and the real world. It can be shown that the proposed method has been implemented and adapted to different objects. The probability of detection for the proposed method can reach 80%, and the trajectory's traceability is more accurate than manual UT. The limitation of the results is that it only considered the object at the assigned location, the object localisation and path planning to the object can be studied in future research.

Keywords: robotic arm; ultrasonic testing; NDT; trajectory control; simulation; reinforcement learning; re-manufacturing; industrial equipment; die and mould; subsurface crack

Personal Acknowledgements

The author would like to give his thanks to his family firstly. It is really, really difficult to do an overseas PhD (Mphil) project and write the thesis during COVID era and along with running a family and doing a full-time job (it is quite like juggling, and not even juggling 3 balls which I can do, it is like 4-ball-juggling, which is my next target, that your hands and eyes are constantly moving). To my wife, who took all the main jobs at home, taking care of the children, I owe you a lot. To my children, Ellie and Cody, you are so good to adapt into the lives in the new country, I am so proud of you. Luckily, with the help from you all, I finally make it to this point to submit the thesis (even unknow of the results). All my love to my wife, my lovely daughter and my cute active son. I love you all. And all my thanks to our parents, who devoted themselves to us all their lives and gave us silent supports which really helps us a lot.

The author would like to thank the supervisor Prof. Xu and associate supervisors Dr. Muftooh, Dr. Ma, for their hard working on the project and patience to wait for me to grow in academic career. Due to my special experience, I cannot contribute to the academic research so much as other PhD students. My supervisors did not blame me, they supported me with less overstatement, and tried to help me at every attempt. I really appreciated that. All the thanks to the colleagues in our research group, Dr Zhang, Dr Liu, Dr Hu, for help me with tackling the details in research and challenges in life. Thanks to Dr Broadway for sharing the ultrasonic testing knowledge and equipment with me. And thanks to Dr Wan, Dr Jia, Dr Pickering for sharing their knowledge and experience.

The author would like to give his thanks to all the partners in EU H2020 project RECLAIM. The supportive work from the colleagues at Aston University.

At last, the author would like to thank all the anonymous science workers, for example, remote control software developers, which make it happen that the remote access to my main PC is possible. Thanks to other ROS developers, which makes the ROS community so informative and accessible. Thanks to all the strangers in life. Some of the strangers have become friends of mine. The known, and unknown strangers are the most common people in life, with the warm moments that the strangers treated us, with all the good work they have done, we can thrive even more. With the kind actions to strangers, we can make a better world.

The main code of this thesis will be shared on BOX, Aston University. If you have any further question about my thesis or any topic that I can help, please contact me at: walterwzz@hotmail.com.

Thank you for reading.

29 March 2024

Stratford-upon-Avon, UK

Table of Contents

Abstract	i
Personal Acknowledgements.....	ii
Table of Contents	iii
Table of Figures.....	v
List of Tables.....	xi
List of Author's Publications.....	xii
Publications related to this thesis	xii
Others	xii
1. Introduction	1
1.1 Background: remanufacturing and maintenance in manufacturing industry	1
1.2 Inspection in remanufacturing and maintenance.....	16
1.3 Motivation of this study	23
1.4 Aims and objectives	25
2. Literature Review	28
2.1 Online/offline NDT in manufacturing industry	28
2.2 Ultrasonic testing in industry	31
2.3 Robotic ultrasonic testing research	35
2.4 Gap in knowledge and contribution of this study	51
3. The Establishment of Simulation Model and Methodology.....	54
3.1 Research methodology	54
3.2 Kinematic and dynamics model.....	56
3.3 Simulation model of robotic arm UR5e.....	65
3.4 Controllers used in simulation	74
3.5 Simulation test and the synchronisation with real robot	78
4. Control System Design and Implementation.....	83
	iii

4.1	Control criteria and strategy.....	83
4.2	Controller design and analysis	84
4.3	Simulation and verification.....	88
5.	Path Planning, Optimisation of Trajectory Planning.....	95
5.1	Path planning for robotic scanning	97
5.2	Implementation of reinforcement learning with ROS.....	100
5.3	Simulation model with RL algorithm	109
6.	Experiment	136
6.1	Equipment and preparation of experiment.....	136
6.2	Transferring RL controller to real-world	142
6.3	Implementation of robotic inspection	145
6.4	Results.....	149
7.	Discussion and Conclusion.....	151
7.1	Discussion	151
7.1.1	Limitations.....	152
7.1.2	Future work.....	153
7.2	Conclusion	154
	Reference.....	156
	Appendix A: Implementation of ROS.....	169
	Appendix B: Codes in This Study	176

Table of Figures

Figure 1 The population of Birmingham city area from 1650 to 2011 ⁴	1
Figure 2 The shareholders of UK corporates gross value-added (1970-2010) ⁸	2
Figure 3 The four waves of industrial revolution and future trend ¹	3
Figure 4 Global use of raw material through the 20 th century ¹¹	5
Figure 5 CO2 emission into atmosphere since 1960 with all key conferences on environment.	5
Figure 6 “4R” strategies used in industry ¹⁵	7
Figure 7 A classic model of die forging processes (link: https://www.forgings.bz/drop-forging/).	8
Figure 8 A classic model of injection moulding processes ²¹	8
Figure 9 Left: The picture of moulds for LEGO manufacturing; Right: mould for shoe sole manufacturing.....	9
Figure 10 Examples of the techniques used in remanufacturing of dies/moulds: (a) GTAW; (b) laser welding; (c) electro spark; (d) cold spray; (e) DED ²³	14
Figure 11 The main processes of remanufacturing.	16
Figure 12 The relationship between inspection, maintenance and remanufacturing ⁶⁵	17
Figure 13 (a): internal defects developed during operation; (b): milling cutter removing surface material; (c): detection of the internal defects; (d): subtractive manufacturing to repair the defects; (e): finish the part with additive manufacturing ⁷²	19
Figure 14 Flowchart of remanufacturing processes on dies and moulds.....	22
Figure 15 (a): illustration of close die forging. (b): the die used in this study as the object. (c): the process of subsurface cracking developing to the surface ⁷⁷	24
Figure 16 Comparison of online and offline NDT methods.....	31
Figure 17 The number of publications on UT applications in different industries in every decade.	32
Figure 18 The historic diagram of UT NDT methods.	33
Figure 19 Different types of UT NDT methods.	33
Figure 20 Literature keyword network of UT on Scopus database.	35

Figure 21 Main types of COBOTs with repeatability feature and number of publications.....	37
Figure 22 Typical examples of robotic UT research in prior literature.	39
Figure 23 Keyword network of literature on Scopus database on robotic UT inspection.....	41
Figure 24 Control types of robotic contact tasks.....	42
Figure 25 The similarities between MDP in RL and robotic control.	48
Figure 26 The numbers of research on typical middleware used for robotic research.	50
Figure 27 Overall research strategy of this study.	53
Figure 28 Coordinate system assignment of UR5e.	58
Figure 29 Gazebo simulation of UR5e.....	66
Figure 30 Planners in RViz.	67
Figure 31 Joints and links in Robotiq gripper (left), and the Moveit setup (right).....	67
Figure 32 Picture of the real robot and the simulation model in Gazebo.	68
Figure 33 Screenshot of collision happened in Gazebo and the collision detection.....	68
Figure 34 Screenshot of force/torque sensor plug-in in Gazebo simulation.	69
Figure 35 Screenshot of the effort controllers used in this study.	70
Figure 36 Screenshot of the pick-and-place task.....	71
Figure 37 Screenshot of the computer vision task.....	71
Figure 38 Simulation model of robotic arm and transducer holder.....	72
Figure 39 Screenshot of the environment in Gazebo simulation.....	73
Figure 40 Force and torque measurement in Gazebo simulation.	73
Figure 41 Constant force control on flat surface scanning in real robot environment.	74
Figure 42 Controllers of the robot used in this study.	75
Figure 43 Max_effort and PID setup for effort controller.....	76
Figure 44 Details of the solvers in the controller.	78
Figure 45 Left: Installation of the ‘external control’ firmware into the teach pendant. Right: terminal display when synchronisation started.....	79
Figure 46 Synchronisation of the simulation model and the real-world robot.....	79
Figure 47 The processes of orientation control of point contact scenario.....	80

Figure 48 The surface scanning on flat surface in simulation (left) and real world (right).....	80
Figure 49 The surface scanning on a curved surface in simulation (upper: simulation environment; lower left: location of the airplane wing on the table; lower right: the curve of contact force during scanning).	81
Figure 50 Flow chart of the whole work processes in the surface scanning task.....	81
Figure 51 Control strategy of compliance controller.....	84
Figure 52 Illustration of virtual spring, damper in force control and target position in position controller.	85
Figure 53 Target wrench rostopic publisher.....	87
Figure 54 Working diagram of RL controller and compliance controller.....	88
Figure 55 Screenshot of working scenario in Gazebo.....	89
Figure 56 Screenshot of rqt tools in ROS.....	90
Figure 57 The end-effector contacts the surface of table and compliance controller started to function.....	90
Figure 58 The surface scanning on the flat table surface.	91
Figure 59 Force curve of the end-effector during movement.....	92
Figure 60 Force curve during the transition phase.	93
Figure 61 Photo of real arm using compliance controller.	93
Figure 62 Force curve for compliance controller of real arm.....	94
Figure 63 Network of literature on orientation optimisation of robotic arm.	97
Figure 64 The raster path planning on wing surface in simulation	98
Figure 65 The raster path planning on wing surface in simulation.	98
Figure 66 The circle path planning on die surface.	99
Figure 67 The surface contour of the die surface, left: trajectory points on die surface, right: coordinates of contour.	99
Figure 68 The implementation of RL in the flow chart.....	101
Figure 69 Structure of implementing the trajectory optimisation algorithm using RL.	102
Figure 70 Structure of openai_ros package realising RL in ROS.	103
Figure 71 Screenshot of UR5e in Gazebo simulation using RL algorithm.	105

Figure 72 Simulation result for the RL algorithm.	108
Figure 73 Training result of the RL agent reaching target.	108
Figure 74 Simulation environment for flat surface scanning.	110
<i>Figure 75 Diagram of PPO algorithm</i>	113
Figure 76 Diagram of A2C algorithm.	114
Figure 77 Diagram of neural network in A2C.....	119
Figure 78 Force curve during one time-step in simulation.....	124
Figure 79 Force curve comparison between compliance controller vs. simple proportional controller.	125
Figure 80 Simulation environment for surface scanning using RL.....	126
Figure 81 Model of airplane wing and the normal orientation vectors.	126
Figure 82 Fuzzy in RL controller.	127
Figure 83 Fuzzy controller.	128
Figure 84 Training processes for curved surface.....	129
Figure 85 Reward curve of RL training on curved surface.	129
Figure 86 Orientation difference between actual and target.....	129
Figure 87 Screenshot of the tensorboard evaluation of curved surface RL training with “done_reward=1000”.	130
Figure 88 RL+ compliance controller+ raster scanning on curved object in Gazebo.	131
Figure 89 Accuracy of RL+compliance controller on curved wing scanning.....	131
Figure 90 Accuracy of normal compliance controller on curved wing scanning.....	132
Figure 91 Simulation environment of real scanning scenario with transducer holder and die.	133
Figure 92 Real world (left) and simulation environment(right) of transducer holder and robot.	133
Figure 93 Full set of the transducer holder in Gazebo simulation environment.....	134
Figure 94 Normal vector of the die surface.	134
Figure 95 Accuracy of scanning on die surface.	135
Figure 96 Hardware used in simulation and experiment in this study.	136

Figure 97 Ultrasonic transducer V203-RM and data collection equipment EPOCH 650.	137
Figure 98 Ultrasonic transducer MR201 and MR203 with equipment EPOCH 650.	137
Figure 99 The structure of transducer holder.	138
Figure 100 The picture of the 3D printed holder installed on the gripper.	138
Figure 101 The picture of the transducer holder with the transducer and cable.	139
Figure 102 Test object for the experiment.	139
Figure 103 The robotic arm UR5e used in this study and the gravity component at the FTS.	140
Figure 104 The measurement of contact force and torque of the end-effector of real robot.	141
Figure 105 The end-effector before and after the orientation optimisation.	141
Figure 106 Comparison between with/without gravity compensation.	141
Figure 107 Illustration of meta-learning.	144
Figure 108 Settings of EPOCH 650.	146
Figure 109 Calibration of the transducer.	146
Figure 110 Screenshot of the software GageView Pro (left). USB and MicroSD card interface on Epoch650 (right).	146
Figure 111 Screenshot of the data and curve of the saved data from EPOCH650.	146
Figure 112 Screenshot of the robotic UT on flat surface.	147
Figure 113 UT raw data of thickness measurement around the surface of the die.	147
Figure 114 Photo of experiment scenario with Epoch, PC, Robotic arm and transducer.	148
Figure 115 Screenshot of scanning experiment.	148
Figure 116 Scanning of complicated surface of the die.	149
Figure 117 Top: “nine dots” ROS logo. Bottom left: logo of ROS Noetic Ninjemys, bottom right: screenshot of Ubuntu 20.04 system.	169
Figure 118 Base environment and created Robot_ur5e_ws environment in Conda.	169
Figure 119 Joints and links of UR5e robot.	170
Figure 120 Example of TF tree of the robot.	171
Figure 121 Screenshot of a launch file.	172
Figure 122 Main structure of ROS.	173

Figure 123 Structure of ROS_control (Source: ROS wiki).....	174
Figure 124 Screenshot of Moveit setup.....	174
Figure 125 Ultimaker S5 3D printer used in this study.....	175
Figure 126 Screenshot of physics parameters setup of the object (left) and the gripper(right).	176
Figure 127 Screenshot of controllers in the config.....	176
Figure 128 Settings of controllers in launch file.	176
Figure 129 Screenshot of RL training code.....	177
Figure 130 Config of RL training.....	177

List of Tables

Table 1 Comparison among remanufacturing methods on dies/moulds.	15
Table 2 Comparison of industrial UT methods.	34
Table 3 Summary of prior literature on robotic UT research.	40
Table 4 Comparison of control methods in robotic control applications.	46
Table 5 DH parameters of UR5e robot.....	57
Table 6 Comparison of target and actual positions of end-effector.....	62
Table 7 The comparison of the control parameters of compliance controller.	92
Table 8 The criteria parameters of RL simulation comparison evaluation.	114
Table 9 The results of simulation tests with A2C algorithm.	116
Table 10 The comparison of results with different hyperparameters in A2C algorithm.	122
Table 11 The comparison of inspection results.	150

List of Author's Publications

Publications related to this thesis

Wang, Z., Zhang, M. and Xu, Y., 2022, September. Development of a robotic arm control platform for ultrasonic testing inspection in remanufacturing. In 2022 27th International Conference on Automation and Computing (ICAC) (pp. 1-6). IEEE.

Wang, Z., Ma, X. and Xu, Y., 2021, September. Weighted multi-element synthetic aperture focusing technique algorithm of ultrasonic non-destructive testing on machinery. In 2021 26th International Conference on Automation and Computing (ICAC) (pp. 1-6). IEEE.

Wang, Z., Xu, Y., Ma, X. and Thomson, G., 2020, September. Towards smart remanufacturing and maintenance of machinery-review of automated inspection, condition monitoring and production optimisation. In 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) (Vol. 1, pp. 1731-1738). IEEE.

Others

Wang, Z. and Wei, C., 2023. Human-centred risk-potential-based trajectory planning of autonomous vehicles. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 237(2-3), pp.393-409.

Zhang, M., Amaitik, N., Wang, Z., Xu, Y., Maisuradze, A., Peschl, M. and Tzovaras, D., 2022. Predictive maintenance for remanufacturing based on hybrid-driven remaining useful life prediction. *Applied Sciences*, 12(7), p.3218.

Amaitik, N., Zhang, M., Wang, Z., Xu, Y., Thomson, G., Xiao, Y., Kolokas, N., Maisuradze, A., Garcia, O., Peschl, M. and Tzovaras, D., 2022. Cost modelling to support optimum selection of life extension strategy for industrial equipment in smart manufacturing. *Circular Economy and Sustainability*, 2(4), pp.1425-1444.

Zhang, M., Sharma, V., Wang, Z., Jia, Y., Hossain, A.K. and Xu, Y., 2023. Online Big-Data Monitoring and Assessment Framework for Internal Combustion Engine with Various Biofuels. *International Journal of Automotive Manufacturing and Materials*, 2(2).

Hu, Y., Zhang, M., Liu, C., Xu, Y., Wang, Z. and Jia, Y., 2022, September. A simulated annealing hyper-heuristic algorithm for process planning and scheduling in remanufacturing. In *2022 27th International Conference on Automation and Computing (ICAC)* (pp. 1-6). IEEE.

1. Introduction

1.1 Background: remanufacturing and maintenance in manufacturing industry

The industrial Revolution happened in the 19th century in the United Kingdom (UK). With the development of propulsion machines, the manufacturing industry, e.g., the textile and iron industries, achieved another level of production capacity due to the machine. The gross domestic product (GDP) per capita in Europe grew over two times after the Industrial Revolution, from \$2,513(1820) to \$7,741(1900) ². This data shows that after the revolution, the average production capacity of every person in Europe, where the Industrial Revolution took place, grew rapidly. Take the United Kingdom as an example, the share of UK manufacturing in the world increased 2 times during 1750-1900 period ³, which shows the contribution of the industrial revolution. From 19th century to 20th century, the manufacturing industry has reshaped the lives of people in the UK, and also the city development of the UK. For example, the metal trade in Birmingham motivated James Watt, who invented the steam engine, to move here to develop the steam machine with Boulton and Murdoch ⁴. The Industrial Revolution also impacted the life and culture of the people who lived here ⁵. For example, more and more people were gathering around the city area to get a better life. Birmingham, as a classic industrial city, as shown in Figure 1, the population in 1900 is almost 7 times of that in 1820 because of the Industrial Revolution.

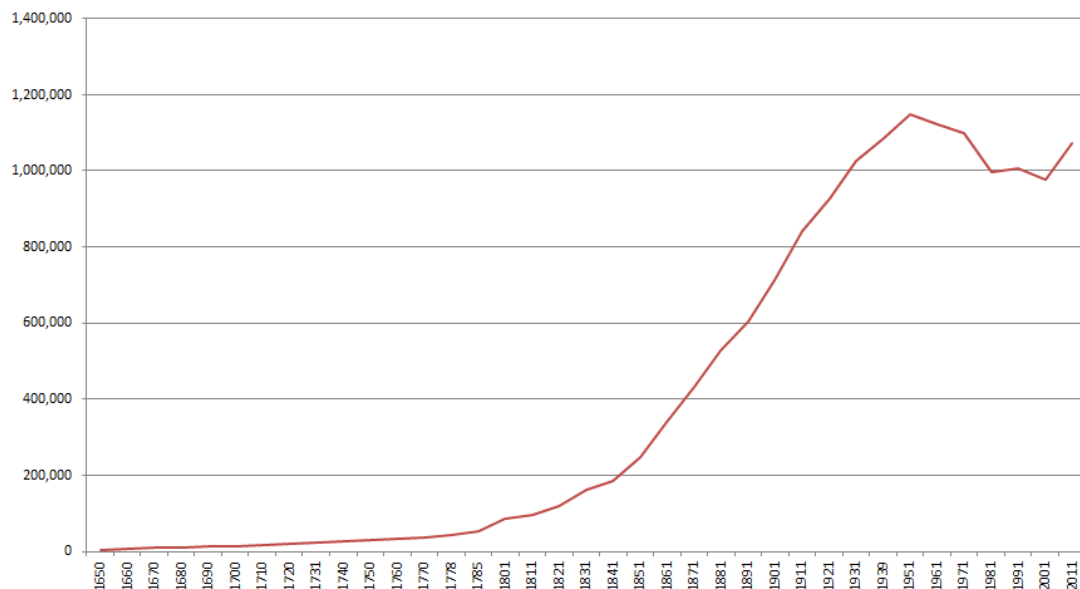


Figure 1 The population of Birmingham city area from 1650 to 2011 ⁴.

The city of Birmingham is chosen as an industry city due to many factors, such as, good geographic features to carry out the logistics in the industry. Natural resource like coal and iron

are around the city, and also the canal and transportation networks benefits the development of industry. And as more people lived in the city area, more and more industries gathered around the city ⁶. Not only textile industry, but also iron making, jewellery making, ship cable and anchor making, etc, which are the main industry of the city. The city of Birmingham played an important role in the new iron and engineering trade during the industrial revolution era. With the modern machinery took more and more parts in the manufacturing, the working condition of the workers were becoming better than before, for example, the working hours are getting much shorter than the hours before revolution ⁷. The working conditions of working humans will still be the focus of the manufacturing industry in the future as it turns into Industry 5.0 era ¹. Therefore, while the Industrial Revolution improves people's living conditions, the working people also improved the level of the manufacturing industry.

As in the 2020s, manufacturing is no longer the largest pillar of the UK's GDP, the focus of manufacturing of UK has also gradually shifted (see Figure 2). The shift can be concluded as the technological change, but also the social, economic and political change ⁸. However, there is no doubt that manufacturing industry is still the key industry in the global economy. As business service and finance sector has more added-value but currently the number of employees in finance sector is only half of that in manufacturing sector ⁹. As more people will choose to work in finance or business industry potentially, the manufacturing industry itself needs swift to adapt the new era (even though manufacturing adds value to the society). Quality, but no more quantity, is nowadays the main focus of manufacturing industry. High-end manufacturing, and high-value-added manufacturing-related industry, sustainable, human-centred manufacturing have been in development and will be the future trend in the UK ¹⁰.

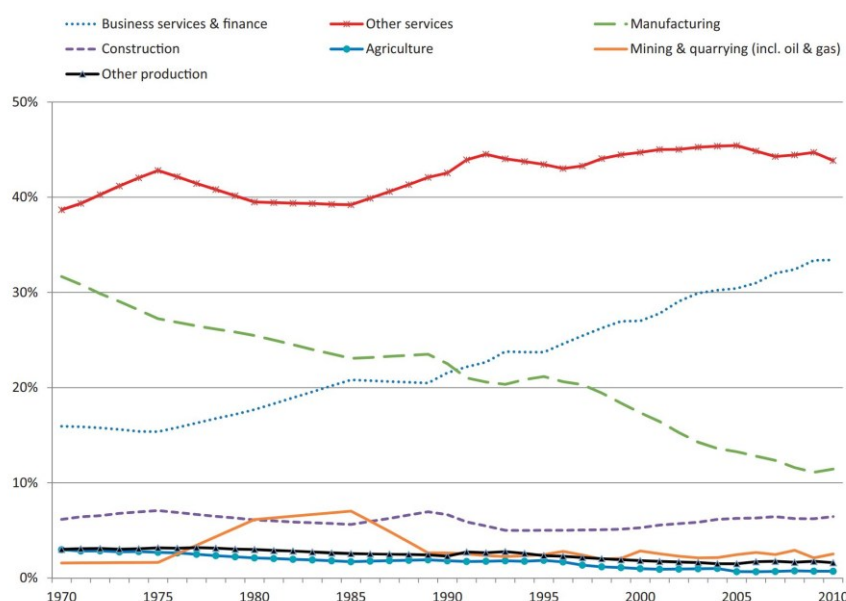


Figure 2 The shareholders of UK corporates gross value-added (1970-2010) ⁸.

Ever since the first Industrial Revolution, there have been three new industrial revolution types, bringing human beings to the Industrial Revolution 4.0 (shown in Figure 3). In revolution 2.0, the modern production line was first introduced by company Ford; the railway and telegraph transferred people and their ideas more quickly than before. Industrial robots and more advanced computing and sensing abilities were introduced in the revolution 3.0. The more advanced network and big data analysis capacity have been developed in modern industry 4.0. More artificial intelligence techniques, such as cyber-physical systems and the Internet of Things (IoT), were developed during this era. The forthcoming Industry 5.0 era, as a return to authentic, more human-centred technologies, will be used to make manufacturing more human-centric and turn the system-centric, technology or knowledge-centric industry into a more human-machine-friendly system, where the machine can better fulfil the creativity of human engineers.

People cannot only sell hamburgers and cut hair for each other; the infrastructure of society and the cost of food and living must come from elsewhere. Manufacturing is the foundation of human lives, and it produces more jobs. Manufacturing transforms people's ideas, designs, raw materials, components, or parts into finished products through various techniques and processes. Basic manufacturing processes such as machining, forming, welding, casting, or assembling can be carried out in a factory or workshop. The goal of manufacturing is to create products that meet the needs and expectations of customers. As the manufacturing technique developed through the revolutions, the manufacturing processes are designed to be efficient, cost-effective, and of high quality. The manufacturing industry is continuously evolving, with new technologies, methods and materials being developed and implemented to improve the efficiency, productivity, and quality. As such, the manufacturing sector plays a significant role in the global economy, providing jobs and generating revenue for businesses and people.

However, with the development of modern manufacturing, more resources have been used in manufacturing to satisfy human's rising living demands ¹¹ (as shown in Figure 4). The unit of y-axis of the figure is the normalised tonnes to the production in the year of 1900. And

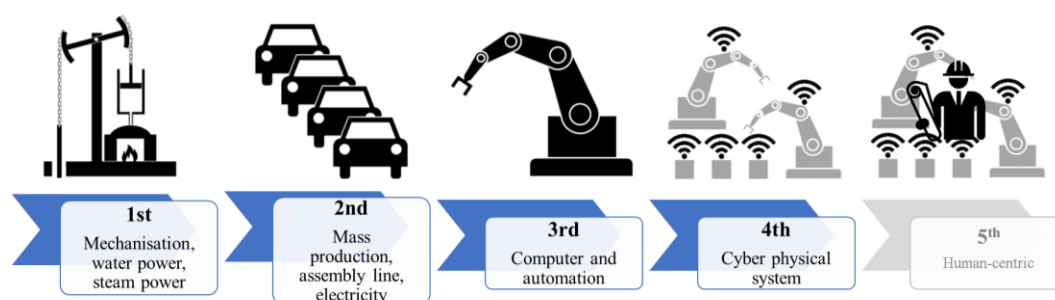


Figure 3 The four waves of industrial revolution and future trend ¹.

according to the report, over 20% of the iron (Fe) production is used in the industrial machinery. Chromium (Cr) has over 50% of its production on metal goods and industrial machinery. Aluminium (Al) has the biggest growth since 1900, the biggest fraction of its use is in transportation and buildings. Since more material is used in manufacturing industry, the situation that the metal resource is running out of stock is catching more people's attention, that the right choice should be made in manufacturing industry ¹². Moreover, manufacturing also produces greenhouse gas emissions, which harms the environment. While the “net-zero” and “carbon-neutral” approach has become the main-stream intention for most countries in the recent decade ¹³, the usage of metal resources needs more effective control. The approaches to these targets are complicated. Since the manufacturers need to fulfil the people's needs and make values, the manufacturing industry is like a large flywheel, which has been started, and it is difficult to stop it. Moreover, manufacturing industry has the responsibility to fulfil the requirements of people. Even in the future Industry 5.0 era, human needs will still be the centre of manufacturing ¹. When people have requirements that have not been satisfied, the factories must make products. While people work for some targets, it is difficult to realise real “net-zero”, for example, people invented bicycles to move quickly, even riding a bicycle is claimed to be “green”, even if the manufacturing processes produce no pollutions, the bicycle itself costs metal resources.

Academic researchers have contributed their wisdom in the direction of “net-zero” and “carbon-neutral” by implementing more advanced technologies, such as IoT and machine learning, which have been used to achieve maximum energy efficiency in manufacturing. Moreover, the technologies can liberate human engineers from repetitive tasks and focus on more critical and lucrative tasks. Besides involving more advanced technologies in the industry, more activities should be done by people, for example, consuming fewer resources, planting more trees to consume CO₂. Despite the establishment of concepts like “net-zero” and impactful international summit conferences and global agreements, the emission of CO₂ never stopped or decelerated (see Figure 5). The predicted CO₂ in the atmosphere will reach 445ppm in 2034, which means CO₂ will be 445 parts per million air molecules. It may not be a significant increase on the number, but it has a bigger chance of limiting global warming to 2°C, which is very dangerous to the environment. Therefore, it is urgent for people to implement more effective and rigorous policies to really improve the state of the environment, especially in the manufacturing industry.

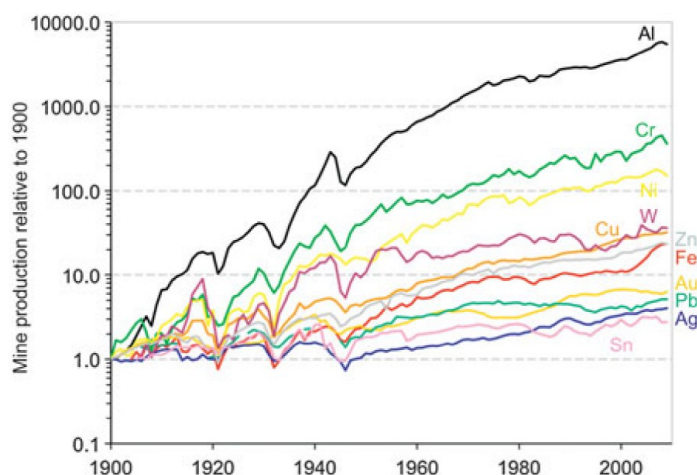


Figure 4 Global use of raw material through the 20th century ¹¹.

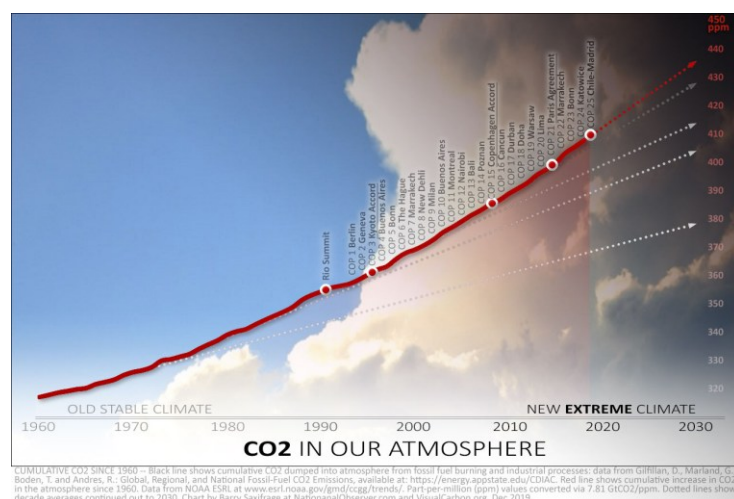


Figure 5 CO₂ emission into atmosphere since 1960 with all key conferences on environment.
(link: <https://www.nationalobserver.com/2019/12/12/analysis/global-climate-summit-cop-or-cop-out>).

Manufacturing industry is not only the foundation of human life, but also the foundation of innovation. Within manufacturing sector, there are normally two basic ways to face the environmental challenges: the first way is to find new methods to tackle emission (for example, post-process for emission from factory) and apply new, more environmental-friendly materials in future. The second way is to consume fewer resources in the current situation. The finding of new materials and new methods to extract metal is the task of material or methodology scientists, meanwhile, more methods have been introduced to reduce current material used in industry. By design, the products can be made with less metal, on the other hand, extending the life of products can be another feasible method. For example, operators used to dispose the broken manufacturing components, e.g., bearings, but now people tend to consider if this is right, if we can extend the use of it, if we can reuse or recycle it. The manufacturing industry is switching from linear economy to circular economy (CE) ¹⁴. Strategies, such as repairing, recondition, remanufacturing and recycling (4R) have been implemented in the industry (shown

in Figure 6).

With these strategies, more and more campaigns have been raised to face the resource challenge¹⁵, for example, 'Make Your Metals Matter' campaign by British metal recycling association, to ask the normal people to recycle more metal resource in their daily lives, such as food and drink cans, foil trays and aerosol cans. It was estimated that 95% less energy will be consumed and 95% less greenhouse gas will be produced if the cans are produced by recycling method. For researchers, there are also many projects regarding 4R in industry, such as, RECREATE (REsearch network for forward looking activities and assessment of research and innovation prospects in the fields of Climate, Resource, Efficiency and raw mATerial, 2014-2018) focusing on policy making and design guiding of recycling actions¹⁶, PREMANUS (Product REMANUFACTURING Service System, 2011-2015) focusing on remanufacturing of the used products¹⁷, RemanPath (Remanufacturing Path Finder, 2018-2020) focusing on guiding medium and small-sized company to take part in remanufacturing, including automotive, sports goods, and manufacturing industry¹⁸. As there are only 2% companies in Europe are implementing remanufacturing, and there is a 90-billion-euro market after 2050, the remanufacturing has a very good potential market. In our EU project, RECLAIM (REmanfaCTuring and refurbishment of LArge Industrial equipMent)¹⁹, the focus is on the industrial equipment which is used for manufacturing other products, for example die forging machine in die forging production.

As shown in Figure 6, 4R is implemented in industry as follows: as the products or parts get broken, the instant repairing will be taken place to make them applicable to be used or sold again. If that is not possible, the key components will be reconditioned to assembly again. During recondition, the goal is to bring the product back to a working condition, often with a focus on addressing specific issues rather than a comprehensive overhaul. A further strategy than that is remanufacturing, which takes whatever steps to make the state of the components as good as new. The product will be re-assembled to realise its function. If all these strategies are not possible, the raw material must be recycled, e.g., shredded into pieces or melted into liquid, to gather all the material to re-produce the product again. The 4R can be all implemented, but the manufacturer can also choose either one to be implemented according to the actual manufacturing situation by considering the feasibility and economic factors.

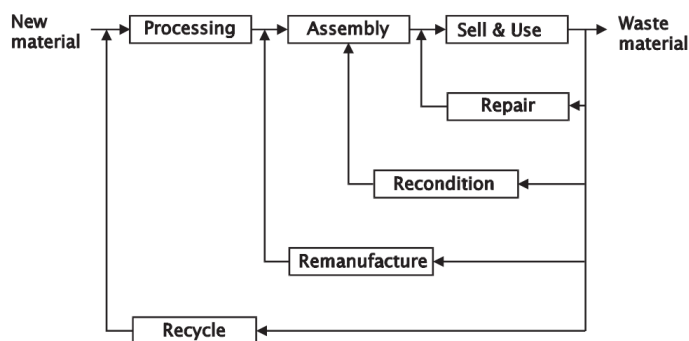


Figure 6 “4R” strategies used in industry ¹⁵.

For each strategy, there are pros and cons, for example, repairing needs the least labour, however, it only fixes the superficial defects of the products or equipment. Moreover, the remaining useful life (RUL) of the repaired product cannot be compared to new products. Reconditioning needs more labour; it fixes the components or sub-components of the products. But it never meets the “good-as-new” state, but better than repairing level. Remanufacturing disassembles the products (also called ‘cores’), it needs more labour to disassemble the product, to implement repairing or reconditioning on the components and sub-components, but it can restore the product to the good-as-new status. Recycling is an environmentally friendly operation, and it creates jobs in collection, processing and manufacturing. These operations need no expertise, but the product must be disposed according to the raw material, then the raw material is extracted from the products, the whole manufacturing processes need to be carried out again, therefore, the economic profit is relatively low. Nowadays, more and more industries start to realise the importance of remanufacturing and commence to use it, for example, the digital device, such as mobile phones. There are many remanufactured products on digital device market.

But while more research is on the 4R of the resource of mass products, less researchers focus on the equipment used in the manufacturing industry. To manufacture more products, many types of industrial equipment are used to improve production efficiency, such as important machinery components in a production line, like dies and moulds. Dies and moulds are both used to cut the raw material or form the shape. The use of dies and moulds has a long history and can be used for manufacturing metal, plastic products. Ever since the stone age, the metal production has started. In the 1400s, the production of metal became a big development due to the invention of the rolling mill by Leonardo DaVinci. During the industrial revolution, the technique of metal production developed, and the products of metal manufacturing became finer. Dies are used to cutting extra materials out in drop forging processes (see Figure 7). The metal is pre-heated and positioned on the lower die, after the drop of the upper die, the metal billet can be formed into the shape of the dies. Dies can be currently used to produce finer

products with new materials, i.e., aluminium and titanium, in the automotive and aerospace industry²⁰. Plastic and composite components can also be manufactured by dies. In die casting, the terminologies of dies and moulds are used interchangeably.

The difference between die casting and injection moulding is the product material, while it is high pressure injected molten metal in die casting and plastic in injection moulding. Mould-making can find its trace over thousands of years. The injection moulding has developed rapidly ever since the Industrial Revolution to produce plastic parts²¹ (shown in Figure 8). The first injection moulding machine was invented in the 19th century; the method was introduced to manufacture more accurate products, such as glass frames. With the development of moulding techniques, mould can be used to manufacture high-valued medical and electronics products. The dies and moulds are used to produce all kinds of products, for example, bottles, shoe soles, and precise high-quality products, such as, LEGO, etc (shown in Figure 9). Even though additive manufacturing (AM) on metal and non-metal materials developed fast these years, dies and moulds are not replaceable due to their rapid manufacturing speed and relatively low cost.

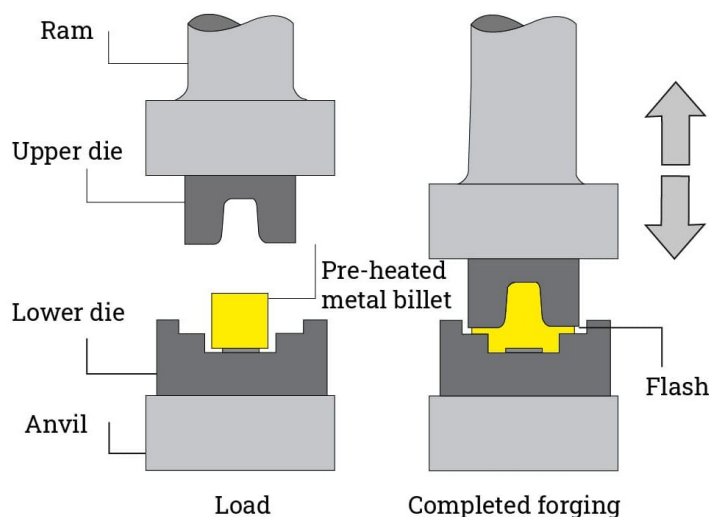


Figure 7 A classic model of die forging processes (link: <https://www.forgings.bz/drop-forging/>).

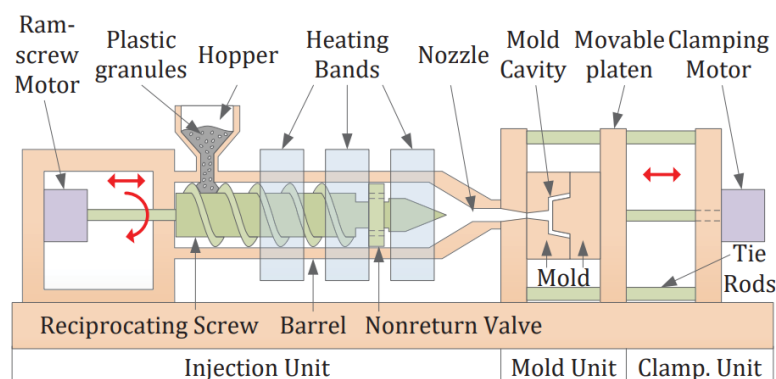


Figure 8 A classic model of injection moulding processes²¹.

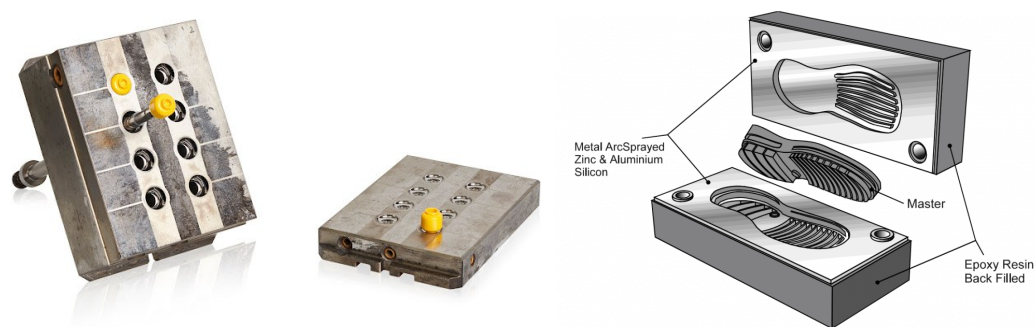


Figure 9 Left: The picture of moulds for LEGO manufacturing; Right: mould for shoe sole manufacturing (link: <https://www.newelementary.com/2018/08/lego-minifigure-moulds-how-are-they-made.html>; <https://www.metallisation.com/applications/metalspraymouldsintheshoeindustry/>).

There are some reasons why a component in manufacturing machine, such as, dies and moulds, should be regularly maintenance and remanufactured. At first, the product is durable and has high added-value; if the restoration or repairing methods are effective; products have the potential to be leased as a service; the technology can last longer than the life cycle of the products²². Cost saving is the biggest benefits of maintenance/remanufacturing on dies/moulds, it was reported the remanufactured part only costs 30-50 % of the cost manufacturing new ones^{23 24}. Regular maintenance or remanufacturing can prevent unexpected breakdown, save downtime. The cost of making new dies or moulds can be saved for the manufacturer. Additional reasons include when the manufacturing of dies/moulds has stopped, or the dies and moulds are high-value and limited manufactured, the maintenance, remanufacturing on the dies and moulds will be necessary. Some dies and moulds may have intricate designs or specialized features that are not easily replicated. Remanufacturing allows manufacturers to preserve the knowledge embedded in the original tooling, ensuring continuity in production processes. Moreover, the maintenance/remanufacturing of dies/moulds extends the lifespan, improves the quality of manufacturing. It also saves the material used for new parts, which has a positive environmental impact by saving overall material resource usage. It was reported that repairing and reusing worn dies can save material use by up to 50%. The material scrap rate during manufacturing can be also improved²⁰. Remanufacturing also provides a potential opportunity to customize or modify dies and moulds to accommodate changes in product design or manufacturing requirements. This flexibility supports the adaptability of tooling to different production scenarios. Considering the reasons above, the maintenance and remanufacturing of dies and moulds is reasonable. According to report, over 80% moulds used in automotive industry undergoes repairing or remanufacturing²⁵.

The state of these types of industrial equipment is important to the quality of products. To maintain the working condition of these equipment, regular maintenance should be carried out. According to reports, the poor maintenance reduces the industry's overall capacity by 5% to 20%²⁶. Currently, the inspections on dies and moulds are mainly visual inspection and

dimensional evaluation ^{27, 28}, which can be implemented before and after the maintenance according to the handbook. But it is not sufficient and may cause some problems when invisible defects are ignored.

There have been serious cases caused by defective dies and moulds. For example, Mylan's EpiPen manufacturer, Meridian Medical Technologies company recalled their products ²⁹. The root reason for this recall is a defective mould, which cannot be checked during normal maintenance but has a subsurface defect causing the products size problem, resulting the pens cannot inject proper amount of epinephrine.

In another case, Toyota recalled over 6 million vehicles due to a malfunction in the power window switch. The cause is a defective mould with subsurface defects ³⁰. Tesla had a massive call back in 2018 regarding the bolts in the power train, which was caused by a series of complex factors, but the defective mould manufacturing it is an important reason ³¹. The problem was solved after Tesla changed the material of the bolts and changed design. Another case is also about automotive parts, a manufacturer of automotive parts was experiencing sink marks on the surface of injection-moulded parts ³², specifically on the interior parts of a vehicle. The sink marks were affecting the aesthetics of the parts and causing customer complaints. The investigation found that the sink marks were caused by subsurface voids in the mould. The voids were formed due to improper venting and cooling of the mould, which led to uneven cooling and contraction of the part. This resulted in subsurface stresses that caused the sink marks on the surface of the part. In 2019, Ford motor also recalled 1.5 million vehicles due to defective dies in the manufacturing of a cable bushing of the transmission, this caused the transmission went into wrong gears and caused accidents ³³. Customer goods manufacturers also suffer from failure cases caused by defective moulds. In this case, the company recalled their product because the handle was easily separated from the product due to a moulding problem during manufacturing ³⁴. A mould manufacturing company Kyodo used UT to inspect their products ³⁵. In the aerospace industry, Pratt&Whitney had an accident in 2016, in which a defective plastic part caused by defective dies during the manufacturing ³⁶.

Therefore, the maintenance of industrial equipment is crucial for the manufacturers, even if the defects in the equipment are invisible. During maintenance, 4R operation can be implemented as a solution to the defective components. For the key industrial equipment that is not damaged to the disposal level, remanufacturing is very suitable since it helps to recover the state of the cores as "good-as-new". Then, the RUL of the remanufactured component can be reliable and further extended. Maintenance methods have evolved since humans started to use tools. Breakdown maintenance is the most traditional method, which means the maintenance only takes place when equipment breaks down. Scheduled (time-based) maintenance is a better

solution than a fixed time schedule, which is not dynamic and does not consider the condition of each equipment. Preventive maintenance, on the other hand, predicts and schedules the future maintenance according to the actual condition of the object³⁷. Condition-based maintenance (CBM) is a more dynamic method which takes online monitoring information to make maintenance decisions³⁸. More advanced method, such as, predictive maintenance (PdM) has been developed. In PdM, the condition is monitored by sensors, and the future RUL is estimated using an artificial intelligence algorithm, and the maintenance plan will be implemented according to the plan.

The maintenance of dies and moulds nowadays is commonly implemented by operators; only manual cleaning and visual inspection are carried out during current maintenance. Many die and mould manufacturing companies, such as Buderus, and Uddeholm, mentioned that the products all experienced UT before delivery. ASTM A578, EN 10228-3 are the standards of UT on steel plates as raw material of dies and moulds. The tooling steel block was tested under the harsh condition as dies and moulds with UT to detect discontinuity³⁹. Cracks and subsurface cracks can be found on the injection moulds made of metal materials, such as, steel⁴⁰. Acoustic emission (AE) has been used to detect internal defects in the injection moulds⁴¹. The reason why using NDT methods, such as AE and UT, is that the dies and moulds suffer from internal defects that are invisible from visual inspection. Moreover, when implementing visual inspection, the dies and moulds are under normal room temperature, the defects can be easily neglected by the operators. While NDT methods can detect the defects under the surface of the cores. Subsurface defects can influence the performance of dies and moulds which will lead to failures in products⁴². With the development of online monitoring, some researchers are using tool condition monitoring (TCM), such as vibration signals, to assess the states of the tools⁴³. Some automatic inspection machines have been designed for the customised moulds, the machines will have the moulds installed on them and move like the actual working scenarios⁴⁴, which can only inspect the working status of the moulds but cannot detect potential defects. A patent was designed by Boeing company to inspect the mould surface using phased array technique⁴⁵. A more detailed review of inspection methods applied to industrial equipment will be introduced in the next section.

With the development of manufacturing techniques, more moulds and dies are made using AM techniques. AM technique has many advantages in making complex parts like dies/moulds. At first, the manufacturing processes are much simpler in AM than in conventional manufacturing (CM). Therefore, the time consumed in manufacturing is shorter. Since the preparation of dies/moulds is very complicated, including prototyping, try-out, design confirmation and final manufacturing. The time-saving in prototyping and manufacturing can save time consumption and overall costs. It is reported that manufacturing of dies/moulds using AM can save over 80%

energy⁴⁶. However, due to the complex parameters of the operation, defects can happen during processes. If the adhesion between layers is not good, delamination will happen. If the cooling is not even, the tool will warp during manufacturing. Metal, plastic, ceramic and photopolymer materials can all be used to manufacture dies/moulds. Metallic dies/moulds are the most commonly used, plastic dies/moulds can be used to produce medical devices, toys⁴⁷. Ceramic dies/moulds are brittle, and it is challenging to produce a large-size ceramic tool. Some of the latest research also used plastic, such as polylactic acid (PLA), polyethylene (PE), polypropylene (PP), to partly or fully make tools on metallic parts, which usually suffer from internal defects⁴⁸. The AM methods can not only used to produce whole dies/moulds, but also to repair or remanufacture them⁴⁹. For internal defects like this, ultrasonic testing and other types of NDT for internal defects are used in the inspection on the tooling.

When the industrial equipment is broken down, the option of the end-of-life recovery/disposal strategy is very crucial to the manufacturer, the consumer and the environment. Even though only a small part (17%) of manufacturers realised this problem⁵⁰. If more broken parts are disposed of, the cost of the manufacturer will rise, and the indirect price paid by the consumer will increase. Moreover, more resources are wasted due to the disposal of new parts production, and pollution and greenhouse gas emissions during these processes will also increase the pressure on environmental protection. As an important option of the circular economy, remanufacturing is a trending option for manufacturers¹⁹. Since remanufacturing saves the processes of producing new parts and recycling old raw material, according to prior literature^{51, 52}, remanufacturing saves 80% of resources, 80% of costs and reduces 85% of the air emissions compared to producing new products. From the economic aspect, it was reported that the remanufactured automotive products could be sold for 40% less than the new products and the profit is 20%⁵³. Remanufacturing, as an industry, also helps to create jobs such as disassembly, testing operators. Remanufacturing, as a newly emerging technique, also helps sustainability development in the developing countries due to the modern techniques used in remanufacturing, such as, IoT, has minimised the technique gap between developing countries and developed countries. Since developing countries consume over two-thirds of industrial goods, the contribution of remanufacturing is very valuable⁵⁴. The evidence above shows that remanufacturing can be beneficial for all three pillars of sustainability: environmentally friendly, economic, and societal beneficial⁵⁵, therefore, it is a sustainable choice for the end-of-life parts for the manufacturers, which can lead to a positive impact to the environment.

As remanufacturing is often provided to customers to restore the state of equipment and reduce total cost, it can be considered as a maintenance option. Since the state data can be acquired during maintenance, it can be a good information source for the planning of adaptive remanufacturing. Via adaptive remanufacturing, the manufacturer can not only gain economic

and ecologic benefits but also the valuable data of their capital ⁵⁶. The data will help to maintain the performance of the equipment, as a key role in the maintenance. The remanufacturing plus maintenance mode can be a new mode of business. The remanufacturing can assess the equipment, implement the maintenance at the ideal time point with optimal cost, and restore the state of the equipment to “good-as-new” to extend the remaining useful life. Therefore, remanufacturing is an optimal method that can be implemented during maintenance.

The remanufacturing of dies/moulds is special comparing to cores from mass production. The quantity is the first difference, since dies/moulds are usually very expensive, the number of cores like this is usually small comparing to mass products. Since the dies/moulds are in unique shapes, the remanufacturing processes are usually customised. The geometry features of the cores are special and unknown in some cases. Remanufacturing strategies are different from the cores from mass production, repairing with advanced techniques is usually used on the remanufacturing of dies/moulds ²⁵. The advanced technologies in AM and subtractive manufacturing are often hybrid to remanufacture the cores, which is named hybrid manufacturing (HM). The direct AM repairing is rarely mentioned in literature, since the wear region needs machine before the application of AM. The methods have been used in remanufacturing include welding ²⁴, tungsten inert gas (TIG) welding (also named as Gas Tungsten Arc Welding, GTAW), Plasma arc welding (PAW), to laser-beam welding, electro spark, cold spray, to main-stream AM methods, such as directed energy deposition (DED) recently (shown in Figure 10). These advanced methods improve the efficiency of remanufacturing, since the methods get rid of the lead times for producing replacement parts or tooling components. This can be crucial in minimizing downtime for manufacturing processes that rely on specific dies and moulds. Advances in additive manufacturing materials, including high-performance polymers and metal alloys, provide more options for creating durable and wear-resistant components for dies and moulds.

For TIG welding, it is a complicated process to complete the welding in repairing the dies/moulds ²⁴. Not only the information of tool material should be known, the location to be repaired, the material of filler welds and if the pre-heating is needed are all necessary. The whole welding processes should be written down in details to make sure the repairing is complete and secured. As welding can hardly restore the microstates of the cores, it can hardly category into remanufacturing methods.

Moreover, in TIG/Plasma welding repairing, much residual stress is left inside the cores. Laser-beam welding, on the other hand, has both the advantages on welding and a more focused heat affected zone (HAZ). The more concentrated energy density makes laser welding capable for more precise remanufacturing. The researchers also started to turn into other methods, such as

cold spray and electro spark.

Electro spark is an efficient method to restore dies/moulds. It is a method of applying a high-performance coating on specific area of dies/moulds. A high current pulse, instead of heat, is input to the substrate of the core to weld a consumable electrode material to the surface of the metallic core⁵⁷. The material of the overlay can be alloy which is not suitable for the previous methods since they will crack during the solidification period. However, during the period of operation, the temperature of the small area can be 5,000- 25,000 K. Different from electro spark, cold spray applied the coating using a cold method, i.e., supersonic particle deposition. The powder of deposited metal is located on the surface of core, when the powder hit the surface on a supersonic speed, the particles deform, and the structure of the surface will re-form⁵⁸. With the method, the thermal stress on the cores can be minimised so there will be no melting material during the process²⁵.

DED methods, i.e., powder bed fusion (PBF) or wire feed method, have a better heat-concentration area, a higher deposition rate²³. Laser power, scan speed and powder feed rate are the parameters that have been discussed for DED in remanufacturing in prior literature. DED can be used most complex geometric components while other methods require flat surface. Moreover, it can handle in making sharper and smaller deposition and lead to a better metallurgical property after the processes. As most of prior literature only mentioned repairing on dies/moulds, remanufacturing has lifted the requirements of techniques to another level. The best accuracy of DED in remanufacturing of die can be 0.5mm⁵⁹. All the methods mentioned above are listed in Table 1 for comparison.

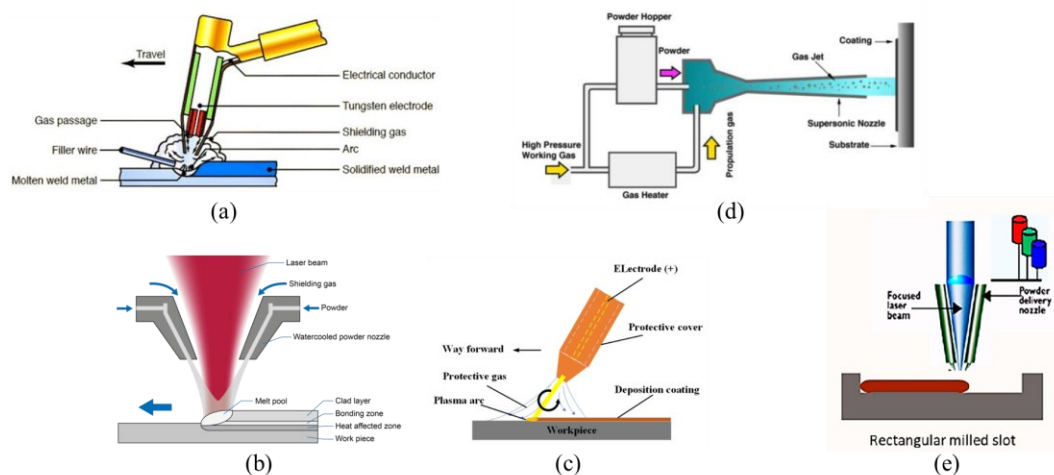


Figure 10 Examples of the techniques used in remanufacturing of dies/moulds: (a) GTAW; (b) laser welding; (c) electro spark; (d) cold spray; (e) DED²³.

Table 1 Comparison among remanufacturing methods on dies/moulds.

Methods	Cost	Metallurgical property	Speed
TIG/Plasma	★★★★☆	★★★★☆	★★★★☆
Laser welding	★★☆☆☆	★★★★☆	★★★★☆
Electro spark	★★★★☆	★★★★☆	★★★★☆
Cold spray	★★★★☆	★★★★☆	★★★★☆
DED	★★★★☆	★★★★★	★★★★★

Although DED can output good properties in repairing, there are still difficulties in the remanufacturing processes, for example, the material compatibility, thermal stress during operation. An optimal remanufacturing may not use only one technique to solve the problem. But comparing to how to implement DED remanufacturing, the more important is where to implement the DED. The location of DED remanufacturing also has a heavy impact on the way to implement remanufacturing and the result of remanufacturing^{23 60}. Especially when the defects locate in the sub-surface of the core, for example, the subsurface porosities and voids appear at sharp transition areas, the preparation work of different damage zones is different. Without precise data of the internal defects, the strategy of remanufacturing may be chosen wrong. The material and tools used in remanufacturing may be wrong and the cost will increase, the downtime will be more than expected. Building strategies are important for the quality of repairing, the features include building sequences, size of removal, build speed, etc. For example, different preparation strategies for repairing different zones for the internal cracks were investigated in prior literature⁶¹. It showed that the trapezoidal removal in the removal-filling strategy had the better overall quality.

Above all, the manufacturing and remanufacturing industrial tools like dies/moulds are complicated. Moreover, either conventional methods or advanced techniques applied on dies/moulds may cause subsurface defects, which are difficult to detect. It is important to detect sub-surface defects as a potential risk. Early-stage detection of subsurface defects and preventing them from developing into fatal surface cracks are very important in the manufacturing industry as a predictive maintenance tool.

1.2 Inspection in remanufacturing and maintenance

Even though remanufacturing is a trending circular business mode, it is challenging to implement it. There are some key processes in remanufacturing: disassembly, cleaning, inspection, operation (repairing, recycling), re-assembly and testing (as shown in Figure 11), the sequence of the processes varies depending on the cores, for example, cleaning may happen before or after disassembly, inspection can take any place in the sequence. There are research challenges within each process in the whole sequence. For simpler dies/moulds assemblies, an overall inspection can be implemented at first to assess the remanufacturability. This process can be called pre-inspection. For the more complicated dies/moulds assemblies, cleaning and disassembly should be implemented before inspection. As an option, regular inspection during maintenance can be used to detect defects for the remanufacturing ⁶². In normal sequences, the equipment or sub-components will be disassembled into single components. If the component is reusable, or it is worthy remanufacturing, cleaning and inspection will be implemented to evaluate the status of the component. Then, the re-processing will be carried out according to the actual defects that the component has; for example, the hybrid manufacturing method (additive + subtractive manufacturing) can be implemented if there are subsurface cracks ⁶³. The equipment will be re-assembled and tested after the re-processing is finished. If the component is not reusable, the possibility of recycling will be evaluated. It will be recycled if it has the recyclability, the raw material will be recycled and rebuilt into new components, and merge into the machinery again. The raw material will be disposed of if it is impossible to

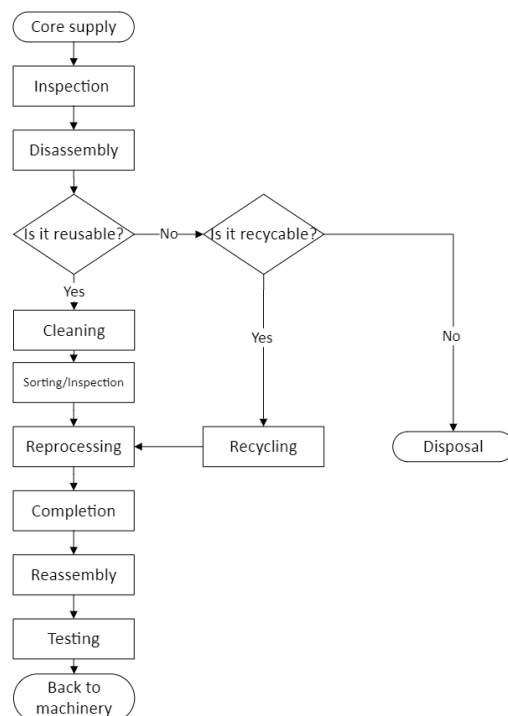


Figure 11 The main processes of remanufacturing.

recycle the parts.

When and how to implement remanufacturing have always been a trending research topic in academics^{64 65}. This problem is similar to the topic of maintenance planning. Prognostic and health management (PHM) technique is the necessary technique that can predict the failure time and schedule the maintenance and remanufacturing. PHM can help make plans, reduce loss with a reliable evaluation of the equipment. In most of the planning research, it is assumed that the inspection is perfect, and all the cores can be graded into remanufacturability levels. According to prior literature⁶⁴, the grading of cores is crucial in remanufacturing regarding the efficiency and cost considerations, so it proved that inspection important for remanufacturing processes. The priority of remanufacturing is various regarding the type of cores, some researchers prioritised the good quality cores⁶⁶. Some found that the grading is more valuable when the cores are in poor quality, however the overall planning of remanufacturing also depends on the cost of the grading⁶⁷. Grading was also found not only reduces the remanufacturing cost but also improve the efficiency of the whole system by reducing the ordering frequency⁶⁸.

The inspection in maintenance can be used to predict RUL of parts, but in prior literature, less focus was on using the RUL to make remanufacturing²⁵. As shown in Figure 12, the relationship between inspection, remanufacturing and maintenance is as follows: the inspection during maintenance can evaluate the state of the equipment and help make schedules for future maintenance. Along the run time of the equipment, the reliability will decrease. Choose the proper maintenance time to implement remanufacturing at a proper “remanufacturability” level will give the manufacturer a better cost level, a better remanufacturability and a better overall value of the equipment. Moreover, the maintenance operations will also extend the lifetime of the equipment.

To evaluate if the equipment is worth remanufacturing, when and how to implement remanufacturing, inspection is the key process before implementing specific remanufacturing

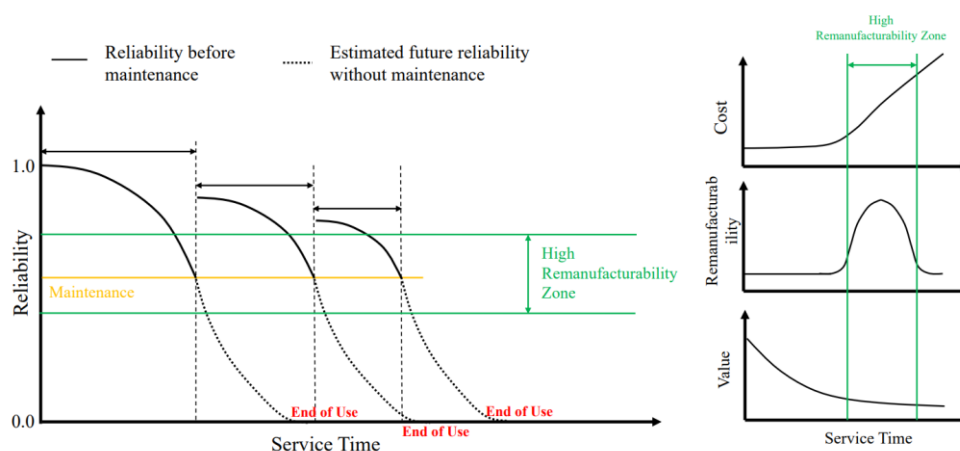


Figure 12 The relationship between inspection, maintenance and remanufacturing⁶⁵.

operations ⁶⁵. Moreover, if HM is applied in remanufacturing, inspection should be also implemented to make sure the remanufacturing quality. Missing the potential defects during inspection will lead to damages. To monitor the status of the components, on-line and off-line methods can be used. The on-line methods monitor the status of the equipment continuously, such as, vibration measurement, acoustic emission and current monitoring. However, these methods can only monitor the status indirectly, the real status of the equipment can be only predicted by collecting data and experience functions. The off-line methods, on the other hand, can evaluate the equipment directly with the contact method. The drawback of off-line methods is that the equipment has to be stopped when inspection. When implementing inspection for remanufacturing and maintenance, the off-line methods are more appropriate, since the equipment has to stop during maintenance, moreover, the off-line methods have more accurate results of inspection. Non-destructive testing (NDT) methods are the most used method in off-line inspection. In the beginning of 19th century, the safety of boilers is crucial. The explosion of a boiler in 1854 caused 20 people died and over 50 people injured, which is the trigger of boiler inspection law in 1864 ⁶⁹. This is the start of NDT inspection.

There are many NDT methods of inspection in industry maintenance and remanufacturing, such as, ultrasonic testing, x-ray, eddy current, etc. In this study, only active inspection methods are considered. Ultrasonic testing (UT) uses ultrasonic waves to propagate through the tested object. It is also called ultrasound, since the frequency is above 20 kHz, which is the upper limit of human hearing ability. The transducer can generate the wave, emit it into the object. And it can receive the reflection of the wave, from wherever there is an internal defect. The ultrasonic waves are motivated via a piezoelectric actuator, which transfer the alternating current (AC) to ultrasonic vibration. The transducer can both generate and receive the waves. The UT methods can be divided into 2 categories, contact and non-contact. For non-contact method, electromagnetic acoustic transducer (EMAT) can generate ultrasonic waves that overcome the inefficiency in the air. Other transducer uses laser to generate ultrasonic waves on the surface of the object. Non-contact is not considered in this study since it is more expensive than contact methods, thus the results are less reliable.

X-ray computed tomography (XCT) is another method to detect internal defects, as it is similar to the medical x-ray, the industrial X-ray needs a chamber to block the beam since the radiation is harmful to the health of human operator ⁷⁰. The cost of equipment is relatively higher than that of UT. Moreover, the application of this inspection also has limitations on the size of the objects. Gamma rays can be used for thicker and larger items.

Eddy current (EC) testing is another inspection method similar to UT. The difference is that EC generates the wave by using eddy current phenomenon directly on the surface of the object.

Therefore, EC is only feasible for the inspection on metallic or conductive products, which can response to the eddy current.

Infrared thermography (ITG) is a new inspection method similar to EC, it can be used for internal defects, however, the object should be an excitation source, e.g., a heating lamp. Active ITG can function on normal object, but the external excitation is needed. Another limitation is that the detection depth is hollow. Computer vision is another widely used inspection method, however only suitable for surface defects and shape defects.

These are the main-stream inspection methods used in NDT in industry. Since other methods all have obvious drawbacks, thus UT is the most common NDT method used in industry, it is chosen to be the method used in this study. It will be used to detect the internal defects in the object for future remanufacturing.

Take internal crack as an example, the location, size of the crack and the development of the crack are important factors to evaluate when implementing remanufacturing. They will help confirm when and which method to implement the remanufacturing, for example, use “hybrid manufacturing” method which includes subtractive and additive manufacturing to restore the state of the core ⁷¹. And how much the work there will be evaluated to make an optimised plan for the maintenance and remanufacturing. Some will question even if the subsurface defects are found, we can wait till the subsurface defects develop to the surface and repair with the same technique, the effort should be the same- the amount of substrate needs to be removed are the same in subsurface defects and surface defects ⁷² (seen in Figure 13). However, it only appears to be the same, the subsurface cracks may develop to extra bigger size according to the working load and the effort remanufacturing it will be different. Remanufacturing a subsurface defect is more beneficial than saving effort, it may save the whole manufacturing system from fatal before the disaster happens, which is more valuable.

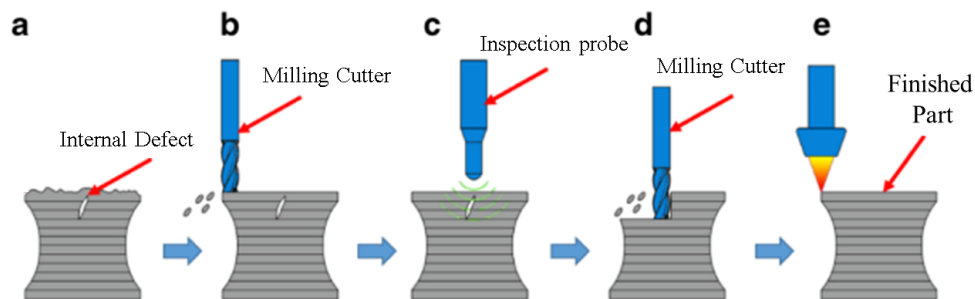


Figure 13 (a): internal defects developed during operation; (b): milling cutter removing surface material; (c): detection of the internal defects; (d): subtractive manufacturing to repair the defects; (e): finish the part with additive manufacturing ⁷².

Key industrial equipment, such as dies and moulds, works in harsh working conditions,

shortening their useful life. Dies and moulds are the equipment that helps to shape products in mass manufacturing according to the design. The use of dies and moulds makes the manufacturing more time-efficient and cost-efficient. They can be used to manufacture products with the material of metal, rubber, glass fibre, polystyrene foams and even food. The dies and moulds can also be used to manufacture complicated products; for example, a simple electronic appliance needs hundreds of dies and moulds. Due to the complexity of the tools, the dies in stamping, for example, can go in the range of 1 to 62 million USD ⁷³, the mould used for the manufacturing of components for automotive interior costs up to 0.5 million USD and 6-9 months to try out. With the development of manufacturing techniques, the emerging technologies have been used in the manufacturing of dies and moulds, for example, advanced CNC (computer numerical control), surface coating, additive manufacturing ⁷⁴. The materials used in dies and moulds are with outstanding strength, for example, 42 HRC- one type of hot work tool steel, D3- one type in the series of cold work tool steels ⁶². According to reports, the value of the market of dies and moulds has reached 26.21 Billion US dollars in 2020 ⁷⁵. Facing the huge value of market, the end-of-life operation of dies and moulds is a big challenge and opportunity. Die and moulding making industry covers a wide range of activities: (a) the manufacturing of new dies and moulds; (b) maintenance, support and remanufacturing of dies and moulds; (c) technical assistance and prototype manufacturing for customers. So, within the maintenance and remanufacturing area, machining tools, such as dies and moulds, is a research topic worth studying.

The most common processing using moulds is injection moulding in manufacturing, e.g., footwear manufacturing, LEGO blocks manufacturing (shown in Figure 9). Rubber, composite and metallic products can be manufactured using injection moulding. In injection moulding, the moulds experience high temperature and high pressure. The thermal change will extend and shrink the surface material of the mould, produce tensile stress between surface layer of material and the subsurface layers. Subsurface cracks will emerge due to the tensile stress ⁷⁶. Dies are the parts used in forging or casting to help shaping the products, e.g., in the manufacturing of doors of dishwasher, body of vehicles. For the dies in die forging and die casting, the material of die experiences not only tensile stress, but also high temperature. The high temperature accelerates the production of oxide of metal, the oxide of metal produces a channel that absorb oxygen from the outer surface, therefore subsurface crack is produced. Thermal shock and tensile stress between dies and the products are the additional reasons for the subsurface cracks ^{77 78}. With all the factors described as above, the subsurface cracks will develop to the surface (shown in Figure 15). Subsequently, the crack will develop into a surface fatal, the quality of product will be heavily influenced by the defective dies and moulds. The tool wear, e.g., the die and mould wear, can not only result in inefficient product quality or economic loss, but also lead to 7 to

20% of total downtime in the whole manufacturing industry. In a survey, a company loses average 300 labour hours and 172 million USD annually due to unexpected tool wear ⁷⁹, which has a big gap to improve ⁷³.

The more accurate the final product is, the higher the requirements for the maintenance of the dies and moulds are. They are not only contributing to the quality, precision, performance of the products, but also the level of the stability of productivity ⁸⁰. The surface quality of the dies and moulds has a big influence on the final products. Therefore, wear resistance, corrosion resistance, and mould releasability are important to dies and moulds.

For the dies, since the tools are under harsh tensile, wear is the most common problem. Die wear leads to surface problems, such as scoring or burnishing damage, and size quality problems. Mould releasability is a parameter for mould, which can be measured by setting a tensile tester between the mould tool and the moulded part ⁸¹. The wear on mould will increase the adhesive on the tool surface, which increases the releasability force. The precision requirements on moulded and died products are normally a few μm . If subsurface cracks develop into surface cracks, the roughness of the surface will change, and the friction of the surface will change, which will affect the quality of the final products. Not only surface roughness but also dimensional accuracy, tensile and compression strength, cost and lead-time can be influenced by the quality of dies and moulds. For example, the die wear can lead to a decrease in the edge quality of the final products ⁸². Therefore, regular maintenance should be carried out for the dies and moulds, and remanufacturing as an end-of-life option is reasonable for expensive tools.

As the additive manufacturing technique developed, more researchers are using additive manufacturing methods to implement remanufacturing on dies and moulds. Hybrid manufacturing (subtractive and additive manufacturing) is used to not only manufacture the dies and moulds, but also to fix the broken surface of dies and moulds ^{83 84} (as shown in Figure 13). Removal operations, such as milling, were used to remove the bumper part and corroded surface, and then laser cladding was used to build the shape layer by layer. The subtractive manufacturing process not only removed the extra material but also prepared the finished surface for additive manufacturing. Some other researchers took the thermal stress and material deformation during laser AM process as a disadvantage and chose other methods, such as cold spray and electron beam welding (EBW) ⁸⁵. The whole remanufacturing processes are similar; milling removes extra material, and then cold spray finishes the repairing process.

To implement these techniques on remanufactured cores is complicated, high precision is required during these processes ⁵⁹. It is even more important to get information on the targeted defects, such as internal cracks. The size, shape and location of the subsurface cracks are key

parameters to carry out the remanufacturing processes ⁸⁶, i.e., evaluating the possibility of remanufacturing, removing surface material and implementing additive manufacturing. NDT inspection, such as UT, can help achieve the related information without breaking the object. By identifying the defects' size, shape and location, it is helpful to confirm the possibility of remanufacturing, the cost of remanufacturing and the methods used in remanufacturing. The possibility of remanufacturing, or named remanufacturability ⁸⁷ (as shown in Figure 12) is the evaluation made by an expert or decision-making system to decide whether the remanufacturing should be worth carrying out or not.

The size, shape and location of internal defects affects the fatigue strength model of the core ⁸⁸. There have also been experience functions based on the size, shape and location of internal defects to calculate the remaining useful life length of the component. Compared to other factors, the location of the internal defects is reported to be more important ⁸⁹. The location of the defects affects the fatigue strength, it is also important to remove material from the right place rather than removing more material. Moreover, with accurate information on the size and location of the defects, the remanufacturing operations can be implemented only on the “region of interest” but not destroy the other part of the object. The remanufacturing operation can be planned precisely, for example, when to implement remanufacturing and how much material should be removed from the specific area of the defect.

The overall remanufacturing processes on dies and moulds are shown in Figure 14. During

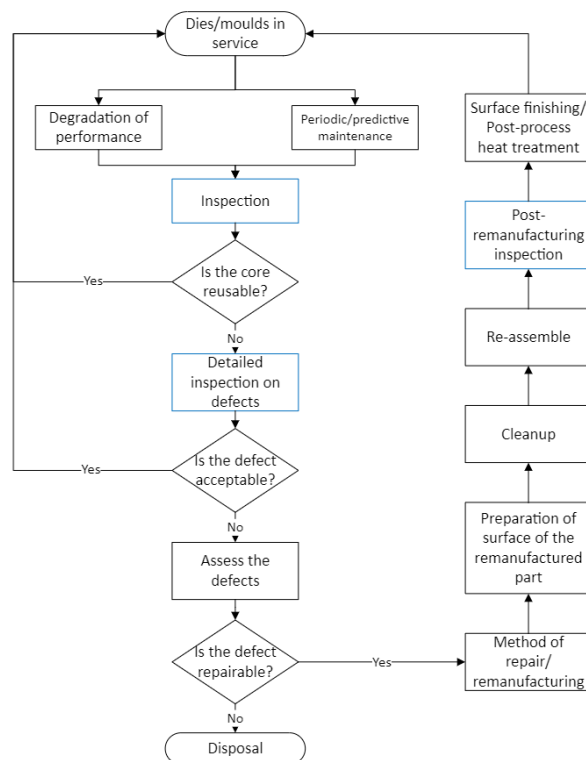


Figure 14 Flowchart of remanufacturing processes on dies and moulds.

periodic maintenance or break-down maintenance, an inspection should be carried out to confirm the state of the components. The evaluation will be made during the inspection to decide whether the die/mould should go back to service. If a defect is detected, the defect should be identified and evaluate to decide if the defect can be repaired in the economic and technical perspective. If the defect can be accepted, the parts can go back to service. If the tool cannot be remanufactured, it will be disposal. If the defect cannot be accepted and needs to be repaired/remanufactured, the core will go to remanufacturing processes. The remanufacturing processes include conventional method, such as welding, and also modern method, such as hybrid manufacturing. The difference between the remanufacturing of dies/moulds and other components is that the surface requirements of dies/moulds are higher than normal parts. Therefore, extra surface finishing processes, such as heat treatment, are necessary before the remanufactured part goes back to service. Re-assembly and testing before service are needed as well. After the remanufactured dies/moulds go back to service, the circular economy of the tools closes its loop.

It can be seen inspection is the start of the loop, however, it has been also reported that the inspection of industrial dies and moulds is normally difficult. The information of the cores, i.e., the original information and current states may be unknown^{90 91}, the material, shape, surface of the cores may vary. Even when the cores is known, the operation of inspection is not easy, since the size of dies and moulds used in industrial is normally too large for the scanning electron microscopy (SEM)⁸². And the detection of subsurface cracks normally depends on SEM. Moreover, these inspections normally need the dies and moulds to be disassembled from the machine, which costs not only human labour, but also precious downtime. The research on the relation between die and mould wear and the degradation of the quality of final products should be studied, and the efficient, online NDT method should be developed to detect the early stage of die wear. This is not only to save cost, but also to save the waste of resource from the final products and the manufacturing of dies and moulds.

1.3 Motivation of this study

Since dies/moulds are important in manufacturing, and remanufacturing benefits the environment and economy, this study focuses on a rarely studied research topic: the inspection of manufacturing equipment, such as dies and moulds, in remanufacturing. Ultrasonic testing (UT) in industrial settings is normally operated manually. This study proposes an automated robotic UT method to improve efficiency and maintain effectiveness.

UT can detect internal defects, including subsurface cracks. The dies and moulds are expensive; immersion UT in water is not selected since it may corrode the object. The shapes of dies and moulds are complex, and the size of the phased array is too large for this type of inspection.

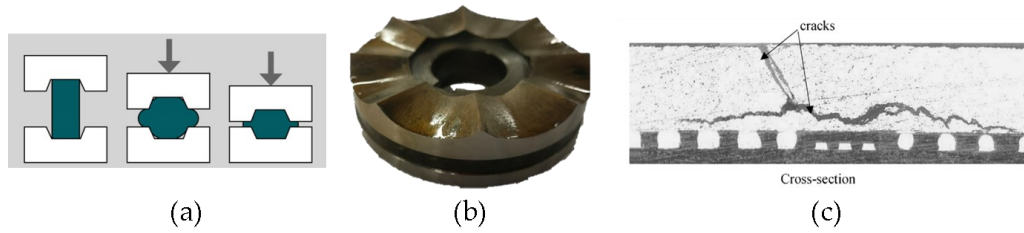


Figure 15 (a): illustration of close die forging. (b): the die used in this study as the object. (c): the process of subsurface cracking developing to the surface ⁷⁷.

Considering other UT methods, they are too expensive compared to our budget. Therefore, a traditional ultrasonic transducer for contact testing is used in this study. The trajectory of the conventional transducer can be planned to cover the object's surface fully. With the amplitude acquired from the conventional UT, the location of defects can be detected.

The orientation of the UT probe is requested to be normal to the object's surface. However, the manufacturing equipment generally has complex shapes (shown in Figure 9 and Figure 15), which increases the difficulty of implementing UT with a robotic arm. Even though dies and moulds are made of strong materials, such as tool steel, H13, P20, and D2, aluminium alloy, AISI4140, 7075 and 6061. The runout of the dies/moulds material during their use in manufacturing can be large; the size wear can be over 1.5mm in experiment ⁹². Another study shows that the size of a die can vary over 1 mm, and the volume change can be over 100 mm³ after 3000 forgings ⁹². The uniform wear value can be over 2 mm ⁹³, a big gap for surface scanning. Most importantly, the probe should contact and be normal to the contact surface of the object to guarantee the best results. Otherwise, the ultrasonic wave will be attenuated due to the incidence angle ^{94, 95}. The volume change after thousands or millions of time operation can influence the UT scanning. Therefore, implementing a robotic arm with an off-line pre-defined trajectory based on the original CAD model is hardly satisfactory for this task. So, the difficulty of the robotic task even increases when robotic UT is applied in remanufacturing.

In contact UT, the probe must contact the object's surface to prevent the ultrasonic wave from scattering and attenuation while penetrating different materials. Therefore, a constant contact force should be applied while implementing robotic UT. While working in a complex environment, the path and trajectory planning of the robotic arm driving the UT probe is also the key. For example, when moving the contact UT probe at the edge of the surface, the moving speed should be adjustable to ensure the probe attaches to the surface.

While integrating collaborative robots (COBOTs) into the study of remanufacturing. In the Industrial Revolution 3.0, industrial robots started to be used in manufacturing. However, the safety and intelligence factors are limited in large industrial robots. Therefore, the emergence of COBOT is a huge step forward to I4.0. With COBOT, humans and robots can work together

more safely. Research on robotic disassembly, testing and inspection with COBOT has increased in recent decades. The number of COBOTs used in industry increased by 23% from 2017 to 2018, and the number is still increasing. In the background of I4.0, more techniques are implemented on COBOT, such as big data, cloud computing, 5G network, Internet of Things (IoT), and artificial intelligence (AI). It is not only COBOT that is used in industry, but the more advanced control algorithms are also implemented in robotic applications.

A Cobot such as UR5e has a built-in force/torque sensor that can acquire 6DOF force/torques. To simplify the equipment used in this study and reduce the budget, only the F/T sensor is used to perceive the environment. Other prior literature uses the combination of F/T and other sensors, such as computer vision; however, using other sensors is not appropriate in this study. This will be justified in Section 2.

For the complex dies and moulds, most inspections must be carried out by disassembling the dies and moulds to make offline inspections⁹⁶, which will lead to longer downtime. If an online inspection method that can detect internal defects without disassembling the tools can be realised, this will improve the inspection efficiency. UT, as an offline NDT method, can be applied as an online NDT while applying a robotic arm. The final target of robotic UT will be a complete mobile solution, i.e., the robot can move the UT probe to the target object and carry out the online UT without disassembling the object. This can be studied in the future.

1.4 Aims and objectives

The aim of this thesis is to propose an automated robotic inspection method for industrial equipment such as dies and moulds. Only a force/torque sensor is used to perceive the environment; no other sensors, e.g., computer vision, are added. Since there is no experience database, a reinforcement learning model will be implemented to learn the optimal actions during training and carry out the actions in simulation and the real world. The proposed method can be implemented in simulation and the real world to verify its robustness.

To realise these aims, the following objectives are to be finished:

1. A compliance controller should be implemented to maintain the contact force between the robot's end-effector and the surface at a constant value $\pm 5\%$ error, e.g., $50 \pm 2.5\text{N}$. The controller should also be able to drive the robotic arm approach to the target position and switch from moving mode to surface-scanning compliance mode.
2. A reinforcement learning algorithm should be able to use the compliance controller as a subsystem to implement automatic robotic scanning. A platform based on ROS should be established to connect all controllers in ROS and the RL algorithms in the `stable_baselines3` library. The training of the RL model should be carried out in the

simulation environment Gazebo. Then, the trained model should be transferred to a real-world experiment to verify its availability. A UR5e robotic arm is used in this study; only the embedded force/torque sensor is used to perceive the environment.

3. The RL algorithm outputs the moving speed, i.e., moving step length, and end-effector orientation adjustment as actions every step after acquiring contact force and current position. The angle difference between the actual orientation and the normal direction on the contact location should be within 15 degrees.

It is hypothesised that the proposed robotic UT method has better efficiency and better POD results than human operators. The improved compliance controller should have a more stable contact force than the standard PID controller. The proposed control method hypothesises that only force measurement is enough to optimise trajectory for dynamic robotic UT. The trajectory in the simulation environment can be transferred into the real world. The RL can help optimise the overall trajectory of the end-effector using the compliance controller. Another hypothesis is that it is assumed that an optimised trajectory can lead to a good UT result. All these hypotheses will be verified in the literature review, simulations, and experiments.

The scope of the research focuses on the simulation and experiment of an RL-based automated robotic UT scanning method. A fixed 6 DOF robotic arm UR5e is used as the manipulator, mobile scanning has not been studied. Though it is an automated method, it only plans trajectory automatically, e.g., orientation, contact force, moving speed, but the moving path is pre-defined (x, y) coordinates according to the shape of the object by the researcher. The object's scope is the dies/moulds used in the manufacturing industry. The object die used in this study is round and has an 80mm diameter. The die is made of solid metallic material, i.e., tool steel, with a reflected surface and curved profile. In the experiment section, the actual die used in the manufacturing is used as the real object. The trajectory optimisation algorithm is applied as a basic controller of the robotic arm, while the RL is applied as the main planner of the overall trajectory. The coding structure and parameters of every controller used in this study are studied. The research employs a quantitative evaluation of orientation error performance and the probability of detection of UT. The scope of the literature review includes the most related literature from 2017 to 2024. The research does not include direct RL training on real objects.

To realise these aims and objectives, the structure of the rest of the thesis is organised as follows: the related prior literature is reviewed in Chapter 2. The inspection methods used in the manufacturing industry and the robotic ultrasonic testing methods are reviewed. According to the results of the literature review, the proposed method is introduced at the end of Chapter 2 to fill the research gap. To fulfil the proposed method, a simulation should be implemented first. The simulation is necessary for safety because the reinforcement learning algorithm tries every

possible solution during training. Moreover, the simulation can be used to verify the possibility of the proposed method before accessing the experiment equipment. Therefore, the simulation is important. The simulation model used in this study is introduced in Chapter 3. The structure of the ROS system, simulation model, and controllers used in this study are introduced in detail in this chapter. The simulation model is also synchronised with the real-world robot. The improved compliance controller will be introduced in detail in Chapter 4, and the design and verification will be carried out in this chapter. The path planning and RL algorithm are implemented in the simulation model in Chapter 5. This chapter will implement the combined controller in the simulation environment. Subsequently, the controller with the RL algorithm is transferred to the real robot in Chapter 6. The related real-world robot experiments are also introduced in Chapter 6. The discussion and conclusion are drawn in the last chapter.

2. Literature Review

To carry out the research on advanced NDT technology on manufacturing equipment is a focused topic for certain area, however, it is also a big topic since either the NDT technique or manufacturing equipment has been developed for over 100 years. The current and historical developments of the technologies should be reviewed to make sure the research can fill the research gap. Since in this study, the topic is focusing on UT on manufacturing equipment, the review of literature will focus on the related academic articles on related topics, such as, ultrasonic testing, NDT, manufacturing, after the year 2000.

2.1 Online/offline NDT in manufacturing industry

There are two categories of NDT in industry, on-line and off-line NDT. On-line NDT can be applied while the industrial operation is still in process, for example, vibration monitoring, acoustic emission, etc. The off-line NDT can be only used when the manufacturing is stopped, e.g., ultrasonic testing, X-ray, etc. All the online and offline NDT methods will be justified in this subsection to find an optimal method for the NDT on die/mould in manufacturing.

For online NDT, the advantages are that the manufacturing processes will not be affected. There are many studies on vibration analysis on cutting tools⁹⁷. However, they can only detect wear when cracking or chipping on the edge of the tools, but they cannot detect subsurface cracks. They can also be applied to dies/moulds, for example, vibration analysis has been used in the monitoring of dies in the nut manufacturing⁹⁸. The vibration analysis study can identify when the faults occur. The frequency of data acquisition can be selected according to the time of operation. The data acquisition does not affect the manufacturing. However, the online NDT can only detect cracks and bends of the dies; it can only give alarms when the vibration mean trend becomes abnormal. Moreover, vibration analysis is a black box, it is difficult to confirm the source of abnormality, especially in a complex vibration system, such as a production line. Vibration analysis is more suitable for monitoring the health of rotating mechanical systems, such as bearings and rotors, but not for detecting early-stage internal defects.

Acoustic emission (AE) is another type of online NDT, which is similar to vibration analysis but is more suitable for crack detection in components. When the cracks happen or any changes happen in the structure of the object, the generated transient elastic waves will be released. AE uses a sensor to collect the reflection of acoustic waves from the internal of the object. Real-time AE can be realised using an optical microphone, this kind of application was used in the detection of internal defects of welding⁹⁹, but this may increase the cost. As AE is a passive NDT, some movement in the object is needed to motivate the acoustic waves. It cannot detect defects when the object is stable, for example, a die part which is not moving. If there is external

sound, e.g., background noises, it may lead to misinterpretation ¹⁰⁰. Besides, AE has a low SNR and a large data amount ¹⁰¹.

Thermography can be an online non-contact NDT in manufacturing. Using an infrared camera, thermographs can be taken, and with comparison, the internal abnormalities can be identified ¹⁰². Passive and active thermography methods have been developed. The difference between these two methods is an internal temperature simulation model. With the simulation model, the normal temperature and the thermal flux with abnormalities can be compared actively ¹⁰³. With the development of technology, laser thermography has been developed to detect near-surface defects. The equipment cost is relatively lower than an X-ray; however, since only one point can be inspected each time, the inspection is time-consuming ¹⁰⁴.

Visual testing (VT), as the most traditional method of NDT, can be used in the manufacturing of different parts and materials, including metal and composite materials. As it is fast and inexpensive, it is normally the initial NDT method in line. With the development of technology, VT has developed direct VT and remote VT, which need external illumination and cameras ¹⁰⁵. However, it has nothing to do with the subsurface defects ¹⁰⁰. It also heavily depends on the expertise of operators. VT can be efficient for product inspection, but it can be challenging to implement VT on manufacturing tools. And even if applied to the manufacturing tools, it can only detect surface defects and shape wear ¹⁰⁶.

Eddy current (EC) can be applied both online and offline. EC can detect the subsurface defects in the hollow depth. However, EC is only limited to the testing of electrically conductive materials ¹⁰⁰. EC has been used in rail inspection while trains are travelling on them, but it was not applied in the inspection of manufacturing tools ⁴⁴. In the recent study applying EC on additive-manufactured metallic parts, EC confuses the surface roughness with the internal defects ¹⁰⁷. The size of the objects under test cannot be too small; otherwise, the ECT probe will be confused by the edge effect, which mixes the edges with the defects. Moreover, the ECT needs a constant air gap between the probe and the objects, but it does not require a couplant, so it is difficult to maintain the constant gap. The smallest void that can be detected using ECT is 0.3mm.

Off-line NDT, on the other hand, needs to move the object components to another place for inspection. Compared to the predicted results from online NDT, the offline inspection results are more reliable, for example, the industrial X-ray in the automotive industry can detect internal defects (which are deeper than subsurface defects) of size 0.03 mm ¹⁰⁸. Though the equipment is powerful, the results rely on the operators' expertise. Since X-ray radiation harms the human body, it is normally implemented in a chamber. The density of the objects also influences the inspection results. The object sizes have limitations, for example, for cylinder

objects, the radius is up to 500mm¹⁰². The sampling process is time-consuming. Besides, the cost of the X-ray equipment is higher than that of other NDTs. Online X-ray inspection can be applied to manufacturing products. There have been online X-ray solutions on production lines¹⁰⁸ However, it is challenging to conduct online X-ray inspections of manufacturing tools, such as dies and moulds.

Ultrasonic testing (UT) is another offline NDT method that requires space and extra time for the UT probe to scan the objects. Like eddy current testing, some studies have used UT to inspect rails online; however, UT is still offline when applied in manufacturing. With some adaption on UT, for example, robotic UT and the application of phased array, UT can be applied half online (inspection implemented on-site, but components do not need disassembly) in the future. With the development of UT technology, many methods have been used in manufacturing, such as PAUT, guided waves, and electromagnetic acoustic transducers (EMAT). UT has proved its capacity applied in turbine blade inspection, it has the potential to apply to the inspection of manufacturing tools, such as dies and moulds⁴⁴. Laser UT has been used for online inspection; however, laser UT is expensive and complex in structure¹⁰⁹ For conventional UT probes, defects of 1mm can be detected. For PAUT probes, defects of 0.2mm can be detected¹¹⁰.

Dye Penetrant Testing (DPT) is another offline NDT since it needs to clean the object's surface, apply a dye penetrant, and then clean and apply a developer. It can only detect surface defects, e.g., cracks. It can detect defect size down to 0.01mm, but only when the defect depth is larger than its superficial open area¹¹¹. Again, the chemical used in PT is harmful to the human body.

Replication metallography testing (RMT) is another offline NDT that needs to be applied to a film on the object surface and then a detailed microstructure inspection of the film is performed. It is only suitable for surface defects. It is used in aerospace and power generation industries¹⁰².

The comparison of each NDT method can be seen in Figure 16. Each NDT method is described as a pentagon, with each vertex indicating the feature of the NDT method (overall detection capacity, i.e., defects depth, resolution), the amount of data needed, complexity, cost, and material compatibility. The distance of each vertex from the centre of the shape indicates the level of each ability. The bigger the pentagon is, the better the method is. After the review, it can be seen that UT is a proper choice for our case study. UT has a relatively robust detection capacity; 0.5~1 mm defect size can be detected. All the other features, i.e., the cost, data amount, material compatibility and complexity, of UT are all on the top of the list. Therefore, UT was selected as the NDT method for inspection implementation in this study. The study of robotic UT control can also be applied in other applications, such as EC inspection or grinding of

manufacturing.

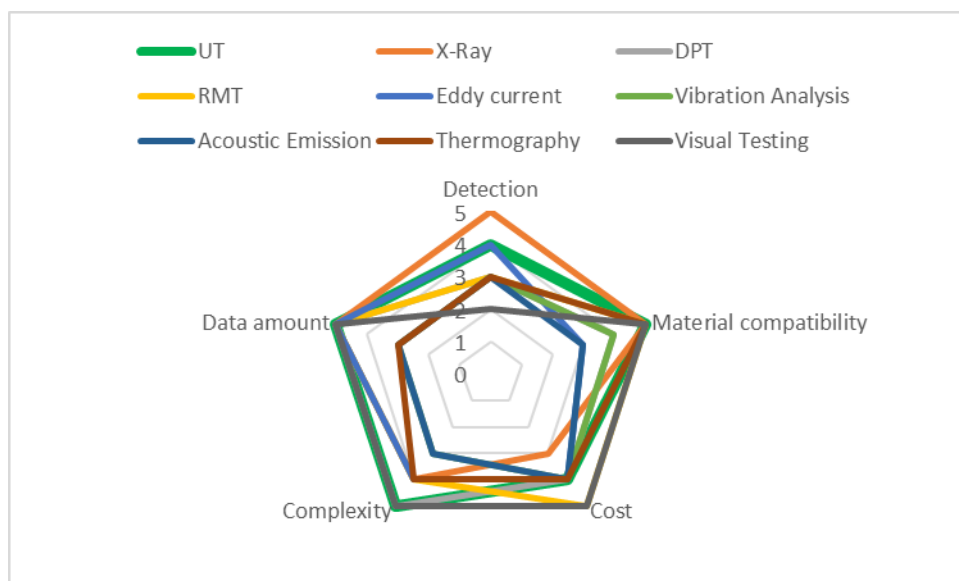


Figure 16 Comparison of online and offline NDT methods.

2.2 Ultrasonic testing in industry

Ultrasound wave was discovered by the biologist Lazzaro Spallanzani in 1794. It was discovered that the bats can navigate their fly without using their eyes, but their ears. The sound wave that the bats use is ultrasound. The frequency of ultrasound is beyond the human hearing range (over 20 kHz), so it is called ‘ultra’. The research of ultrasound was brought after the famous sinking of Titanic in 1912. The researchers started to study for a method to detect iceberg or any underwater obstacles when the visibility is not good. The first published article about ultrasound wave application is the navy application of submarine detection in the first world war ¹¹². The application was quite similar to the original ultrasound function in bats. After the submarine detection project, the French scientist Paul Langevin developed the ultrasonic transducer using piezoelectric effect. Nowadays, the ultrasonic wave can be used in NDT, range detection (installed in automobiles), identification (used for indoor object localisation) and motion sensors (for automatic doors). In this study, the range detection, identification and motion sensor functions are not considered.

Application wise, besides navy use for submarine detection, the ultrasound can be also used as a tool of underwater historic site detection and imaging ¹¹³. The UT inspection was firstly used in medical use in 1930s. The UT transducer with frequency over 3 GHz was used to image internal organs of human. Since it is portable, reasonably priced and not harmful to human comparing to magnetic resonance imaging (MRI) and computed tomography (CT), the UT equipment is now standard for emergency response teams in hospitals. With the development of UT transducer and imaging technology since 1980s, the 3D even 4D (live 3D) medical

imaging is possible for fetus ¹¹⁴. In the industry sector, the UT has been more and more used in nuclear power industry from 1960s, to check the crack in the boilers or pipelines in nuclear reaction plant. The technique of NDT has a huge development during 1970s. Moreover, the emerging of prediction tools of defects boosted the need of NDT. After 1980s, UT became more common in rail, welding, and manufacturing industry. After 1990s, UT has been more used in manufacturing industry. The number of publications with the term “ultrasonic”, “NDT” and related industry were investigated (shown in Figure 17). In 2010s, the number of publications on rail and welding industry accelerated quickly, e.g., crack in rail and welding were inspected using UT ¹¹⁵. The stars in the figure indicate the most increasing number of publications in that category. From these stars, it can be seen the developing trend in each section of research. For example, UT research firstly started in nuclear industry then shifted to manufacturing and aerospace.

The publication number has a strong relationship with the development of hardware. Ultrasonic testing (UT) in NDT has developed for decades. Hardware aspect, the transducer has gained the resolution by evolving from electric conductive transducer to piezoelectric transducer in 1920s. Based on piezoelectric structure, steel quartz steel sandwich structure was developed. Pulse echo instrument was developed in 1940s ¹¹⁶. The techniques helped to improve the quality of UT results, but only amplitude can be achieved. To achieve images directly, phased array (PA) has been developed in 1960s. The principal of phased array is similar to the radar array, multiple ultrasonic transducer elements are combined to emit and receive ultrasonic waves in the probe. For phased array, advanced algorithms, such as, synthetic aperture focusing technique (SAFT) (1970s. 1980s) ¹¹⁷, total focusing method (TFM) ¹¹⁸ (2000s), have been

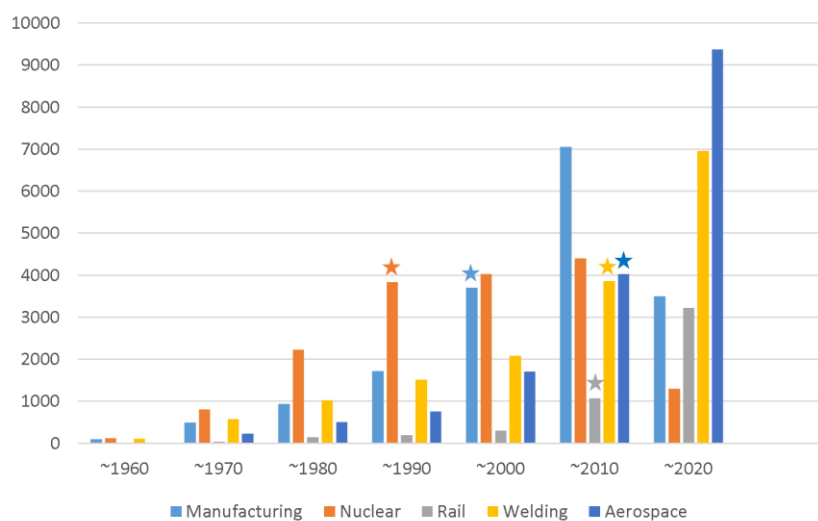


Figure 17 The number of publications on UT applications in different industries in every decade.

studied ¹¹⁹. However, since PA is much more expensive on the hardware, it is difficult to use PA in academic research. moreover, the image achieved by PA can be acquired by single element transducer. Time-of-flight diffraction (TOFD) is another type of UT which was invented in 1970s and developed in 1990s. It is good at measure the size of cracks and widely used in welding industry. Air-coupled UT filled the needs that the couplant is not necessary in some application scenes, such as, wood products and composite plates. In the late 1980s, the high-frequency air-coupled UT can be compared to contact UT. Immersion test has been used in research since 1970s. This method has been integrated with other types of UT, such as, PA, TOFD. The historic diagram of each UT method is listed in *Figure 18*. The curves in the figure show the number of publications in each decade of every method. The number of publications of each method has been converted to the same level to fit the figure. The single-element number is missing since it is not mentioned in all literature, it is difficult to achieve the number

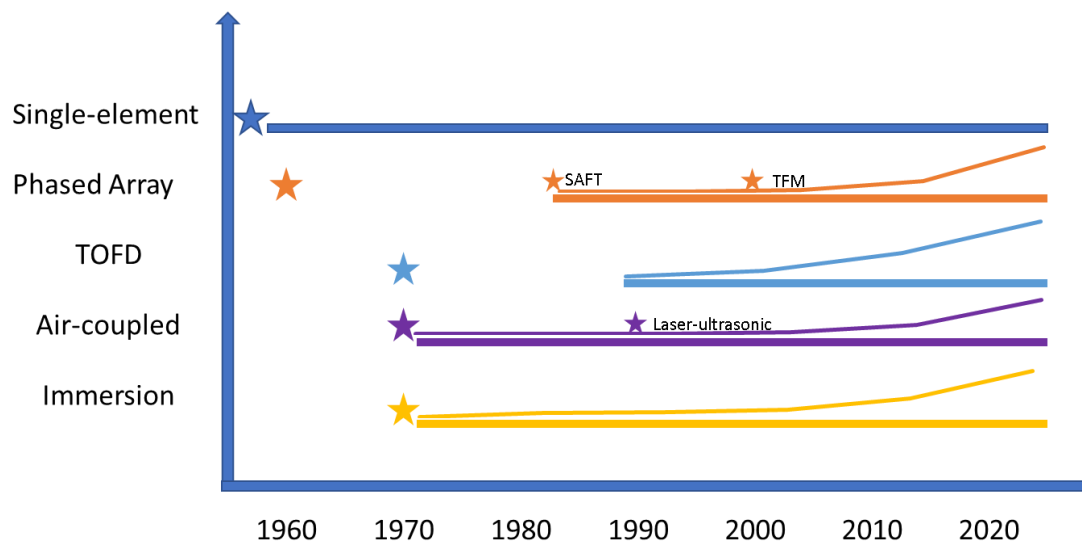


Figure 18 The historic diagram of UT NDT methods.

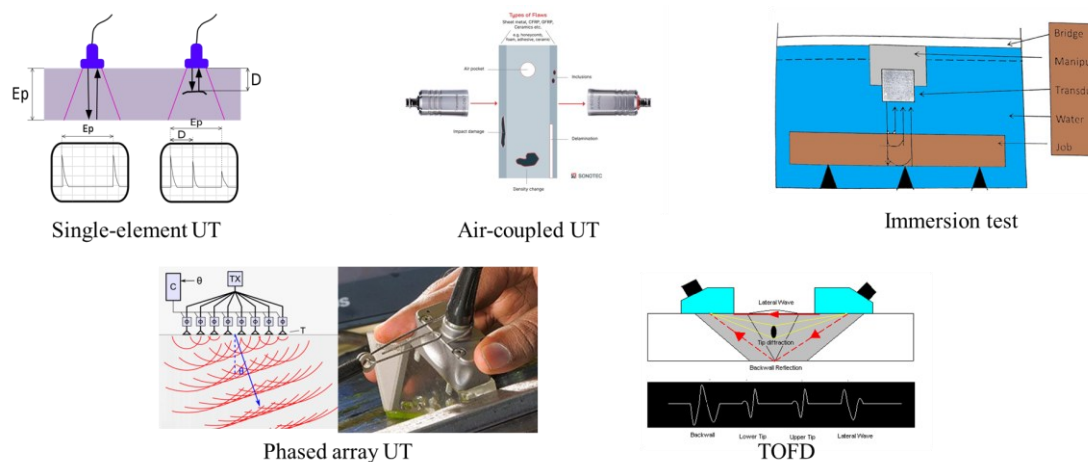


Figure 19 Different types of UT NDT methods.

of publications specific on single-element UT. The stars in figure indicate the development of each method. It can be seen that the development of impactful methods has the effects on the UT related applications in industry.

The comparison of the advantages and drawbacks of different UT methods is listed in Table 2. The illustration of each type of UT is shown in Figure 19.

After reviewing the prior literature and requesting quotations from the market, it can be shown that air-coupled UT and PAUT are too expensive to be used in this research. The air-coupled UT also features high processing time and depends highly on the surface finish. Immersion test is not suitable for metallic components, since the object is exposed to water and susceptible to corrosion. TOFD needs two probes, which increase the difficulties of the research. Therefore, Single element contact UT is chosen in this study. Even though single-element ultrasonic transducer is the conventional sensor, the challenge is to integrate the transducer to the modern application and carry out accurate inspection results.

The limitation of single-element UT transducer is that it can only record the amplitude, the imaging of the object can only be obtained by post-processing. Moreover, the post processing of imaging directly affects the quality of result of inspection of the object defects. Some researchers used SAFT algorithm in the post-processing of single-element UT ¹²⁰. Researchers also applied single-element UT in more advanced inspection scenarios, such as the inspection

Table 2 Comparison of industrial UT methods.

Methods	Advantages	Disadvantages
Contact UT	<ul style="list-style-type: none"> • Reasonable price; Small contact area; 	<ul style="list-style-type: none"> ○ Only amplitude achieved;
Air coupled UT	<ul style="list-style-type: none"> • Non-contact; 	<ul style="list-style-type: none"> ○ Expensive; ○ Resolution not efficient;
Phased array	<ul style="list-style-type: none"> • Direct imaging; • High resolution; 	<ul style="list-style-type: none"> ○ Expensive; ○ Big contact area;
TOFD	<ul style="list-style-type: none"> • Good to detect cracks; 	<ul style="list-style-type: none"> ○ Two transducers; ○ Not suitable for complex shape; ○ Object exposed to water;
Immersion test	<ul style="list-style-type: none"> • No need of couplant; • No need to contact; 	<ul style="list-style-type: none"> ○ Water tank needed; ○ Complicated processes; ○ Time-consuming;

of wire and arc additive manufacturing (WAAM) products ¹²¹. The single-element UT probe was installed on a holder at the end of the production line to assess the states of products. The post data processing of the A-scan data from single-element UT can be used to assess the defined defects in WAAM products.

Thus, single-element UT is highly related to the expertise level of the operator, and the cost of UT is very high due to the slow labour processes ¹²². Hence, the integration of machine will increase the high resolution and high scanning speed of UT inspection. Using the single-element transducer to implement NDT requires accurate moving resolution and control, therefore, the robot is an optimal option. According to author's knowledge, the integration of robotic arm and single-element UT transducer is not done by any researcher.

Another figure (output by VOS viewer ¹²³) can be shown is Figure 20, which shows the literature keywords in the Scopus database for articles on UT in manufacturing during 2000-2023. It shows that more literature is focusing on the optimisation of ultrasonic techniques, UT on 3D printing products/constructions, inspection of materials and process optimisation of UT. Therefore, there is still a research gap to fill regarding the UT availability of manufacturing equipment and automatic robotic implementation.

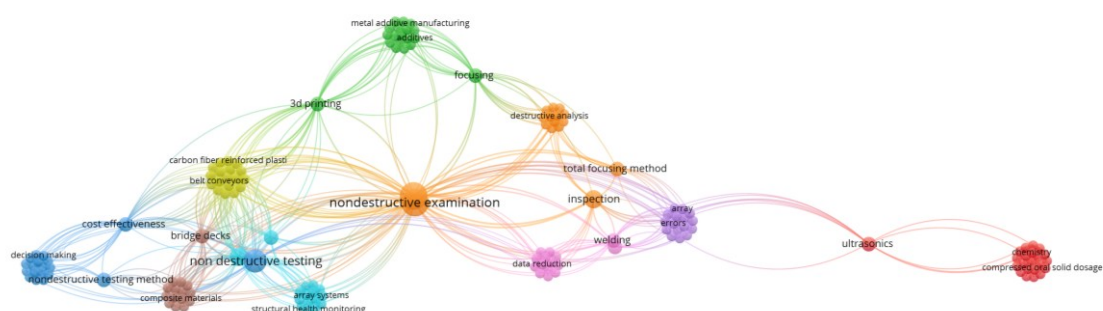


Figure 20 Literature keyword network of UT on Scopus database.

2.3 Robotic ultrasonic testing research

Since most of current UT in industry is still manual, manual UT is lack of the efficiency and effectiveness, depends heavily on the level of operators. For some tasks, more operators are needed to implement the inspection, discuss the results and future plans. Thus, it offers almost no traceability or possibility for further analysis. More researchers are considering robotic application in UT for future application, since it provides efficiency, traceability and reliable data for post processing ¹²⁴.

In the beginning of robotic UT research, most researchers used industrial robots ^{125, 126, 127}. In the research at the University of Strathclyde, a Matlab-based control platform was established

to control the KUKA Agulis robot ¹²⁵. The ultrasonic transducer is a roller dual-element probe. A circular shape steel pipe segment and an aluminium staircase were tested using structure from motion (SfM) to reconstruct the 3D model of the objects. The control of the robot was based on computer vision. However, the focus of the research was on the computer vision reconstruction, the results of UT were not focused, only the success rate of obtaining the ultrasonic measurement was calculated.

In another research from Beijing Institute of Technology, the object, a turbine blade, was fixed on the end-effector of the robotic arm to move. The probe was fixed in the water tank for the immersion test ¹²⁶. The challenge of this “test object grasped by robot (TOGR)” method was to keep a fixed distance between the object and the probe. The proposed method was computer-aided manufacturing (CAM) algorithms. The trajectory planning method was improved to maximise the amplitude of ultrasonic energy, but the details of the method was not mentioned. The TOGR method was also compared with the conventional “ultrasonic probe grasped by robot (UPGR)” method, the results in this case showed the TOGR was better. The TOGR method was innovative, but it is not appropriate when the object is large. Moreover, the object to be remanufactured has a size difference from the original CAD model, so the trajectory planning method is not suitable in inspection of remanufacturing.

A KUKA KR90 was used to move a roller 64-element phased array probe to inspect the artificial defects, i.e., side drilled holes (SDH) to simulate inclusions such as keyholes of lack of fusion defects, in welding of the staircase ¹²⁸. TFM and SAFT algorithm of PAUT were used, the defects can be detected, but the control of robotic arm was not mentioned since the trajectory task is simple, straight line. Since the off-line path planning needs accessory to reconstruct the surface, some other researchers started to study real-time method to reconstruct the surface ¹²⁹. The orientation optimisation was carried out using the real-time UT data.

The industrial robotic arm can finish the robotic inspection tasks, however, since industrial robots are large and only moves according to the codes, this leads to some safety issues, i.e., if an operator is in the path of robot, the industrial robotic arm will hit the operator without stop. When collaborative robot (COBOT) is the trend of robotic research, it is safer and more flexible to use a COBOT in complex tasks, such as, robotic inspection. The advantage of a COBOT is that the safety features of the robot is better than industrial robots. The robot can be stopped when operators block the movement of the robotic arm. Moreover, the COBOT is smaller, therefore, more flexible, faster and easier to install than industrial robot. COBOT is more cost-effective than industrial robot. And COBOT can be applied in more advanced applications, such as, dual arm, external control. Even though COBOT is not capable for heavy loads, its fast response and good flexibility are the features why many researchers use them as the robotic

application.

Universal Robot (UR) is the first company which starts the COBOT manufacturing. It was also said the name of the company comes from the origin of the word “robot”, a novel written in 1920s, Rossum's Universal Robots, by Czech writer- Karel Čapek. Their UR5 is the first type that started the COBOT trend in 2008. Comparing to their competitors, such as, Sawyer, the precision and control of their robot is much better and reliable. Sawyer is a good product, it has single-arm and double-arm options with an interactive touch panel. However, the accuracy and repeatability are much worse than UR. With the new manufacturer of COBOTs, the conventional industrial robot manufacturers also step into the booming market. For example, KUKA has a COBOT type- LBR iiwa, which is used by many researchers. The repeatability is 0.1 mm, which is not as good as UR (0.03 mm), but enough for some tasks. Some other types have their own features, such as, Techman robots have embedded camera; ABB Yumi has single-arm and double-arm options. The current main-stream COBOT types are listed in *Figure 21*. The repeatability and the number of publications on the COBOTs are also listed in the figure. The number of publications was searched using the terms “COBOT”, “industry” and the type of the COBOT. It can be seen that UR5 has the optimal repeatability and the biggest number of publications.

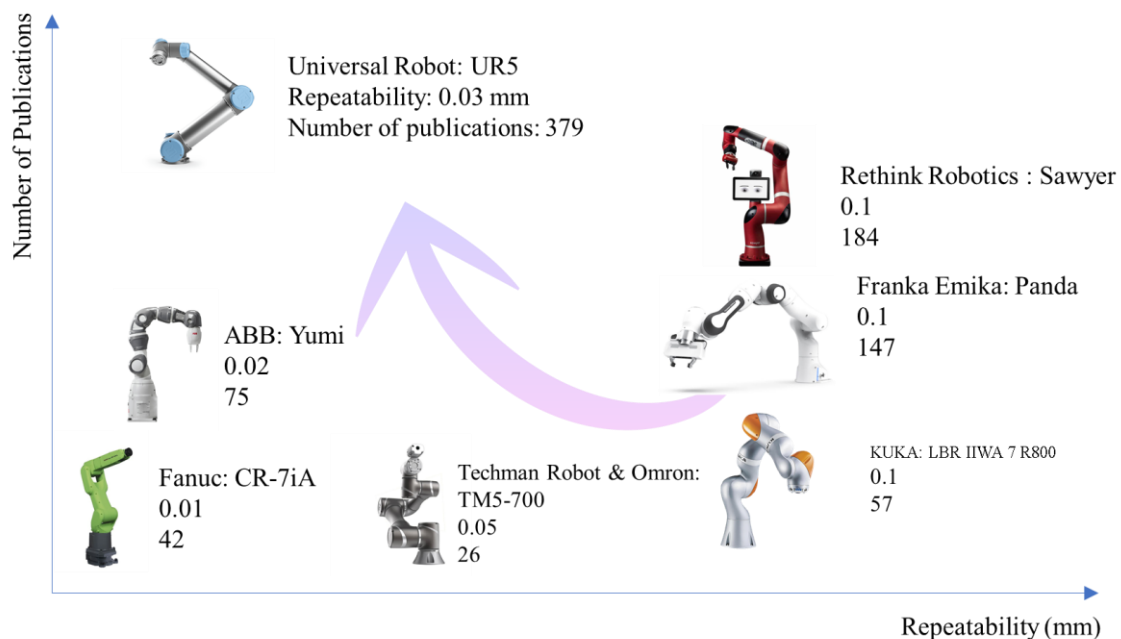


Figure 21 Main types of COBOTs with repeatability feature and number of publications.

In some research, robotic UT using COBOT is used to inspect defects in additive manufactured (AM) products ¹³⁰. In this article, a KUKA LBR COBOT was used to move a roller PAUT probe to inspect AM object. The AM object was produced using an industrial robotic wire and

arc additive manufacturing (WAAM). In the UT part, it only finished the robotic inspection on a straight line WAAM specimen. A contact force of 50N was applied, but the trajectory of robot was only a straight line. To implement UT, the profile of the surface of object should be reconstructed. Some research used computer vision, while other used robotic laser ¹³¹. To guarantee the robotic UT can be carried out in one time, the surface was scanned using the robotic laser scanner. Subsequently, raster path of robotic UT can be generated based on the reconstructed surface. A COBOT was used and ROBODK software was used for simulation. For bigger objects, such as wind turbine blades, some researchers integrated COBOT on the autonomous guided vehicle (AGV) ¹³². With the feature that COBOT has a good robot-human collaboration capacity, the human-robot collaborative NDT was also carried out ¹³³. In this case, robot and human collaborated each other on a welding production line to inspect the quality using UT. COBOT was also used with other type of UT, such as pitch-catch UT ¹³⁴. In this research, the UT transducers were integrated in the gripper of COBOT. The UT was implemented when the object, the oral solid dosage in this case, travelled through the COBOT on the production line.

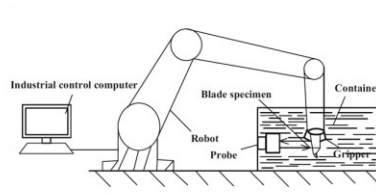
Besides the robotic UT NDT research, the robotic UT in medical research can also bring innovations to NDT research. Researchers from Technical University of Munich used a KUKA COBOT to drive the ultrasound probe to carry out ultrasound inspection on In-Vivo and human volunteers ⁹⁵. For medical ultrasound testing, the probe should also be normal to the object surface, which is human skin and muscle. A model-based method was used to optimise the orientation of the probe according to the contact force measured during the inspection. Due to the geometric feature of medical ultrasound probe and the soft texture of human muscle, many parameters in the function were estimated via regression from experiment dataset. The results were compared with human operators, it showed that the proposed method has more stable performance. The researchers from biomedical school, Tsinghua University, also proposed a solution with a UR3 robot ¹³⁵. The probe was installed directly at the end-effector of the robot. A policy-gradient reinforcement learning algorithm was used to train the agent to achieve an optimal posture and position according to the contact force. Since the scanning was implemented on human soft skin, the simulation was not accurate, so the researchers directly implemented the experiments. Direct implementation of RL on real objects seems unrealistic, which needs further verification. Finally, the proposed robotic method was compared with free-hand method, the results showed that the results only differed 3%. Similar to this method, Lin et al. also developed a robot control for robotic medical UT ¹³⁶. In this study, a tool platform with spring was designed. A rolling ball was installed at the end of the tool platform. The spring and the ball made the calculation more complicated. The method was tested on flat surface, curved surface. However, these two methods did not consider gravity and bias components of

the load (the typical robotic UT examples can be shown in Figure 22). The main research on the research of robotic UT is listed in Table 3.

Industrial Robots:



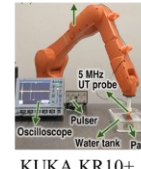
KUKA Agilis+roller probe
(Khan et al., 2021)



STAUBLI TX90L+immersion probe
(Xiao et al., 2017)

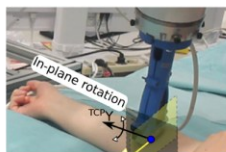


KUKA KR90+PAUT
(Zimmermann et al., 2020)

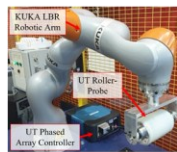


KUKA KR10+
immersion probe
(Mineo et al., 2022)

Collaborative Robots:



KUKA LBR iiwa 7+
Ultrasonix probe (Jiang
et al., 2020)



KUKA LBR iiwa
7+ roller PAUT
(Zimmermann et al.,
2021)



COBOT+AGV+
immersion probe
(Vicat, 2022)



UR3+ pitch-catch
probe (Sultan et al.,
2023)



UR10+ contact
probe (Bakopoulou
et al., 2022)

Figure 22 Typical examples of robotic UT research in prior literature.

In prior literature, there was also research integrating robotic UT with other sensors, such as, Huang¹³⁷ used computer vision combined with force control to smooth the movement during UT. At first, a depth camera was used to reconstruct the environment of the experiment. Two force sensors were installed on both the contact sides of the probe. After the 3D reconstruction of the environment, the normal direction was calculated by three points around the region of interest. The two force sensors helped to smooth the movement of the probe by measuring the contact force when moving. At last, the ultrasonic testing images were collected and reconstructed into 3D model. This was good research but implementing computer vision is not suitable for this study, since the object in this study is mainly industrial equipment. The industrial equipment is made of metal, the surface of the parts is reflective, there will be errors in the reconstruction of computer vision. Although other sensors, e.g., laser sensors can prevent problems like this, but additional sensors increase the overall cost and risk of errors of the system. Therefore, in this study, the proposal is to use only a force/torque sensor to plan for the trajectory of the probe.

In Figure 23, it can be seen that there has been some literature on robotic UT on the Scopus database since 2000. The research topics focus on techniques of automatic inspection/detection/localisation, inspection of 3D printing products, and construction. It can

Table 3 Summary of prior literature on robotic UT research.

Use cases and reference	Advantages	Disadvantages
Industrial robotic arm + roller UT probe 125, 127, 129	<ul style="list-style-type: none"> • A combination of perceptions of the surrounding environment, i.e., computer vision and contact forces. • PAUT used. 	<ul style="list-style-type: none"> • An advanced scanner was used. Only flat surface was scanned. • More budget for facility.
Industrial robotic arm + immersion UT ¹²⁶	<ul style="list-style-type: none"> • object-in-hand inspection. • Non-contact UT. 	<ul style="list-style-type: none"> • Difficult to control. • Water corrosion to metal. • Long preparation time. • More budget for more equipment.
Cobot + AGV + immersion UT ¹³²	<ul style="list-style-type: none"> • Flexibility. • Mobility. 	<ul style="list-style-type: none"> • Only suitable for big objects. • Difficult to control. • Only the flat surface was studied.
Cobot + Contact probe UT ¹³³	<ul style="list-style-type: none"> • Suitable for detailed inspections. 	<ul style="list-style-type: none"> • Difficult to post-process the results.
Industrial robotic arm + medical ultrasound probe ⁹⁵	<ul style="list-style-type: none"> • Use the results of ultrasound image to estimate the orientation of the robotic arm. • Suitable for dynamic environments. 	<ul style="list-style-type: none"> • Not suitable for industrial applications. • Contact force is estimated. • The orientation adjustment is based on table-searching.
Cobot + pitch-catch UT ¹³⁴	<ul style="list-style-type: none"> • Using cobot and UT to tackle internal cracks on compressed oral solid dosage (OSD). 	<ul style="list-style-type: none"> • Trajectory plan not included. • Only simple flat surface objects were studied. • Too many external sensors. Complicated in computation.
Depth camera+ robotic arm+ Ultrasound probe ¹³⁷	<ul style="list-style-type: none"> • Combining perceptions of the surrounding environment. • 3D reconstruction of object. 	<ul style="list-style-type: none"> • Only suitable for medical scenarios. • Orientation calculation according to depth camera has errors.
Research gaps:		
<ul style="list-style-type: none"> - The perception of the environment relies on external sensors, which increases the cost and complexity. - Lack of detailed study on trajectory planning optimisation algorithm. - Lack of combining the overall optimisation method of trajectory during scanning. - Lack of study of robotic UT targeting industrial equipment. 		

be seen that more industrial robotic arm was used, and more robotic inspections were implemented in the aerospace or construction industry. More robotic control was implemented by pre-planned path or computer vision methods, but they are not adaptive enough for the

surface scanning task in this study. Therefore, the research topic in this study, i.e., automatic coordinate robotic arm UT using contact force measurement on manufacturing equipment dies and moulds for remanufacturing, is a research gap.

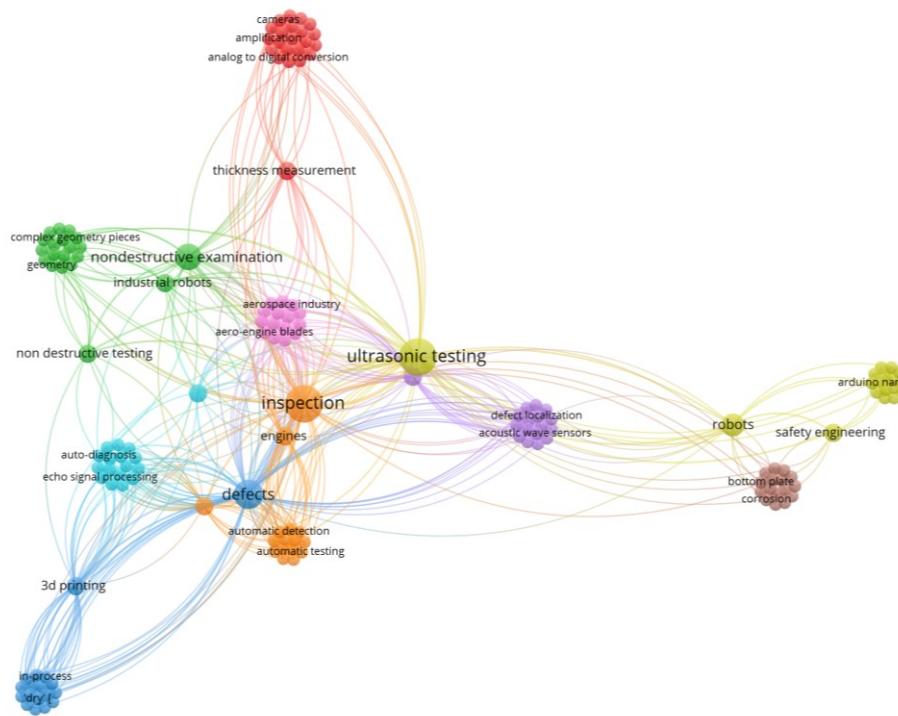


Figure 23 Keyword network of literature on Scopus database on robotic UT inspection.

In prior literature, researchers have developed some methods to optimise the orientation control of robotic arm in contact robotic tasks¹³⁸. There are three kinds of contact robotic tasks, i.e., compliance control, tactile control and hybrid position/force control¹³⁹. Compliance control is for most kinds of robot, in which the motion of the robot on surface of the object is studied. The interaction between the robot and the environment is the studied problem. Compliance control involves adjusting the robot's stiffness or compliance to allow for controlled interaction with the environment. In compliance control, there are active and passive controls. The passive control is to use elastic parts, such as springs, to store the energy to overcome the compliance in the environment. On the other hand, active compliance control is to use controller to drive stiff components to overcome variable compliance actively. For tactile control, it is more focusing on the hand-shape human-like robot doing jobs like solving cubes¹⁴⁰. Besides compliance control, tactile sensors and integration of multiple sensors are the topics in tactile control. Tactile control focuses on using tactile sensors or feedback to make control decisions during contact tasks. The hybrid position/force control is to control position and force simultaneously in a hybrid control in separate control subspaces. It does not only optimise the

interaction between robot and environment, but also drive robot to target position. The contact tasks in this study are in this area. However, traditional position/force control, i.e., model reference control, sliding mode control, and basic learning control, is more dependent on modelling using a fixed parameter controller or controller based on look-up table or functions, which is lack of adaptability and robustness ¹⁴¹, therefore, it needs optimisation on the algorithms in the position/force control. The current main control types of robotic contact tasks are listed in Figure 24.

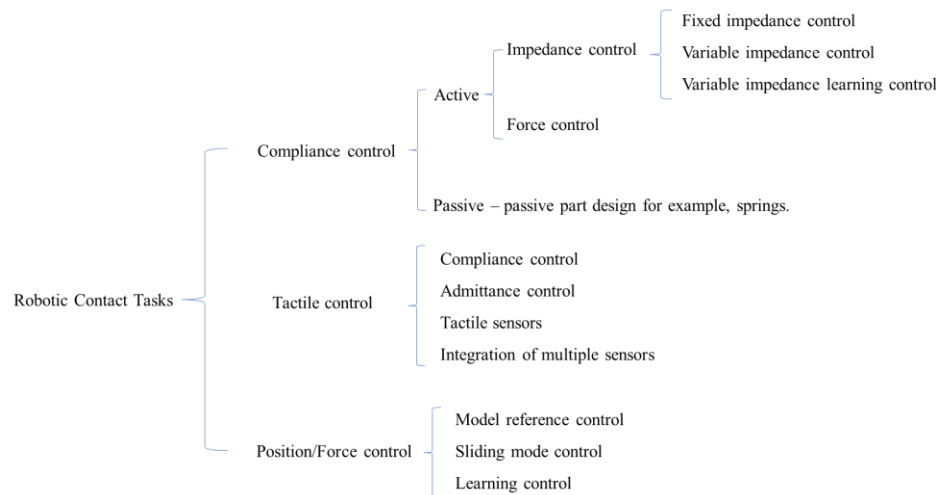


Figure 24 Control types of robotic contact tasks.

Since the task in this study is UT scanning on surface, the position control is more straightforward, i.e., trajectory on a pre-defined routine along the surface. However, the pre-defined path only contains (x, y) coordinates, the z coordinate is dependent on the actual surface and contact force between the end-effector and the surface. The force control and the simultaneous control of position, pose, and force are more focused in this study. To improve the traditional compliance control, more advanced algorithms should be involved.

Impedance control is the most used control method in the active compliance control of robotic problems. In impedance control, there are fixed impedance control, variable impedance control (VIC) and variable impedance learning control (VILC) problems. It is a VILC problem in this study since the impedance during inspection is variable and the status of end-effector needs to adapt along the trajectory based on learning in different scenarios. VIC is only suitable for immediate application of control, such as simple and repetitive tasks, but the parameters need a longer time to tune. Moreover, it is not adaptable when the scenario changes.

Many control methods, such as model predictive control (MPC), Differential Dynamic Programming (DDP), genetic algorithms (GA), particle swarm optimisation (PSO), imitation learning, machine learning, and reinforcement learning, can be used to solve the VILC problem.

MPC is suitable for tasks that have a mature, robust model which can calculate current performance and predict future performance as well. The model predicts the change in the dependent variables of the modelled system that changes in the independent variables will cause. It adapts the parameters by predicting the performance of the next step; therefore, the establishment of the model and tuning of parameters are challenging. There are different approaches, such as the model-focused method and the hybrid method. For the model-focused method, a more detailed model was established for different movements and different scenarios¹⁴². If the movement is in a free space, there will be too much computational load for solving the formulas. As mentioned in the research, the MPC controller can only optimise the controller in the middle level. It needs an accurate model and cannot accomplish overall high-level control. For the hybrid approach, researchers implemented MPC with other algorithms in VILC tasks¹⁴³. In their study, a probabilistic ensemble neural network (PENNN) was established as the model in MPC. Cross-entropy method (CEM) was used to improve MPC performance. With the CEM, MPC has evolved like RL. However, the surface scanning task in this study does not have a reliable model, and the objects can be variable in different scenarios. Therefore, MPC is not suitable.

GA is suitable for the optimisation of control problems, but like MPC, a detailed, robust model is needed for GA. With the model, the chosen parameters can go through crossover and mutation processes to find an optimal parameter setting. The robotic used GA as the control algorithm to write on paper in the latest research¹⁴⁴. Writing is a difficult surface contact task; however, it does not require orientation control. The requested angle on each joint is in the gene arrays. The overall fitness function is calculated based on the joint angle and the distance between the target position and the action position of the end-effector. However, each task needs a different fitness function, and the algorithm cannot be real-time since the calculation load of random gene arrays is heavy.

PSO has recently become a trending optimisation algorithm for robotic arm control. It has been used in the impedance control for dual-arm robot¹⁴⁵. The PSO algorithm is used to adaptively control the impedance when the static error is not available. PSO was applied to search for the optimal parameters in the formula for solving the impedance force. PSO is used to converge the contact overshoot in 0.03s. It has an outstanding performance of ± 0.08 N contact force tolerance. However, only force tracking was studied in the article, and the application was not real-time. PSO is also used in the parameter optimisation in machine learning¹⁴⁶. The contact force is controlled via a PID controller. The impedance controller is controlled by a neural network which needs a joint angle to contact force map. PSO was implemented to optimise the efficiency of choosing the weight parameters in NN.

Imitation learning (IL) is a good way to improve impedance tasks; the methods applying imitation learning have been improved with the development of NN. For example, imitation learning has been applied in the cable assembly manufacturing¹⁴⁷. Since the task needs a higher precision, sensors cannot fulfil the requirements, RL was applied to improve the trajectory. It also developed a self-imitation mechanism to hindsight the policy it learned from RL. However, IL has its limitations, for example, it is not suitable for precision work and not adaptable to different kinds of robots.

Machine learning (ML), or basic supervised learning, is similar to IL, which needs supervision from a human tutor. Instead of getting guidance directly from expert operators, ML tries to derive the optimal trajectory from labelled sample data. Support vector machine (SVM) was used in the training^{148 149}. In the studies, the training data will be labelled into categories, and then the robot will be trained based on the dataset. The tasks using ML are interactions with small objects, such as gripping tasks, which are simpler than surface scanning tasks. Like IL, the results of ML training heavily depend on the human factor, i.e., the human-labelled data and the settings of the NN used in the training model. There is little research using ML to solve impedance tasks; the reason lies that the impedance control has no clear ground truth, for example, how the trajectory should be in a dynamic environment. The labelling of the training data is a big challenge, and it may not lead to a good result.

RL is the current most suitable solution for VILC problems since it avoids the shortcomings of the conventional control methods. In conventional control method, the control depends on the force and position feedback to close the control loop. The parameters in the model-based method will be estimated by using the differential changes of force and position. However, in various impedance control problem, the model of control is normally non-linear, and the controller can only tune the impedance gains by repeating the learning processes.

Iterative learning and RL are the most used method to solve this problem. However, iterative learning is more suitable for fixed tasks, and the parameters of iterative learning controller are mostly tuned manually, so RL is the better choice for variable environment. For example, for the tactile tasks, the exploration on the unknown surface is a trending research topic¹⁵⁰. In the unknown surface exploration task, the information by contact was maximised by RL algorithm. The RL was used to plan exploration path to predict the surface shape of the object. Since this task needs the robot to learn to discover the features by repeatedly interacting with the environments and transfer the function on to new environments, so RL is suitable for this case. Similar to this study, researchers studying industrial processes, such as, grinding, deburring, found it is difficult to extract accurate contour of the surface¹⁵¹. Moreover, the sudden change of the shape will lead to force overload during the processes and lead to a robot failure. Factors

such as, curved surface, insufficient rigidity of robot, the interference of the environment, causes the force to fluctuate. Since the trajectory of the robotic arm on a curved surface is challenging, and current used traditional methods of pose optimisation are lack of accuracy and efficiency on curved surfaces. RL algorithm does not need prior information and can optimise the poses on the real time. It autonomously sets and optimises the parameters in controllers, so it can be used in robotic tasks like this. In their study, RL was used to find the relation between force control and compensation parameters in the force controller.

The artificial intelligence method, such as, reinforcement learning (RL) can be also introduced to control robotic UT ¹³⁵. In this study, RL was used to estimate the torque which can be used to rotate the end-effector. The force/torque measured from the sensor was used as the state. The environment is the flexible environment that the ultrasound probe worked. The total moment of the end-effector was used as the reward function to calculate the actions. RL is a new trending of robotic arm control, but it is more suitable for continuous surface scanning. The control in this study is mainly for normal direction calculation in point inspection. RL can be considered in the future study of this article. In other tasks, such as peg-in-hole assemble tasks, RL algorithms were also used ¹⁵². Demonstration information was used to improve the convergence speed of RL, and long short-term memory (LSTM) network was used to reconstruct the critic network of PPO algorithm. But as a contact task, only visual information was used in this study. As dynamic robotic UT is the variable impedance learning process, therefore, RL is chosen in this study.

The overall comparison of all these algorithms is listed in Table 4. The selection of the control algorithm in the VILC problem depends on the difficulty, complexity, data accessibility and computational constraints. After considering of all these factors, RL was chosen as the control algorithm in this study.

Regarding the robotic scanning tasks using only force/torque perceptions, there is some prior literature studying this topic. For example, some researchers from Stanford University used a robotic arm model in Mujoco to implement peg-in-hole with only force sensor and position information ¹⁵³. In their simulation, the data was collected and then labelled as successful or failed by researchers during scanning. The model will learn from that labelled data. A force sensor is also used in the guided robotic arm operation ¹⁵⁴. However, only human-robot

Table 4 Comparison of control methods in robotic control applications.

Methods and literature	Advantages	Disadvantages
MPC	Accurate for model-based method.	Needs accurate model for each movement. Can be computational expensive when the problem is complex.
GA	Overall optimisation. Suitable for optimisation of one parameter.	Unexpected mutation may happen. Struggling for complex, multi-target problems. Needs different fitness functions for different tasks. Computational expensive.
Swarm optimisation	Model-free approach. Suitable for multi-objective optimisation problems.	Sensitive to parameters. Computational expensive. Not suitable for a dynamic environment.
Imitation learning	Easy to apply. Rapid to learn. Less risks of exploration.	Requires expert knowledge. Visual IL is difficult for robots to learn. The transferred trajectory can be inaccurate.
Machine learning	No expert knowledge is needed. No model is needed.	Need training labelled data. No ground truth for a dynamic environment.
Reinforcement learning	No model nor human factor is needed. The model will learn to solve the problem itself. Suitable for dynamic environment and generalisation of the model.	Training needs trials. Complex settings in training.

controller to optimise contact-based method 3D scanning¹⁵⁵. It only used the force controller and inverse kinematic (IK) formula to reconstruct the surface model of the object. However, the orientation of the end-effector was not considered. In industrial applications, grinding robot control also uses force control to optimise trajectory planning¹⁵⁶. The tooling path was also pre-defined. Since grinding application is more complicated than UT scanning, it removes material from the surface. A material removal model was established based on the contact force, spindle speed, etc. Except for the removal model, a force plan is pre-defined for the grinding process. They also implemented the simulation to verify the processes, but in a mathematical way, not a high-fidelity simulation environment. Another difference is that the tool orientation does not have to be normal to the surface, only the contact force in the normal direction must be calculated in the removal formula, while the orientation is more important in UT. Overall these prior studies proved that only force control can implement the trajectory optimisation.

In medical UT, only-force method has been used in some studies^{95 157 158}. There are researchers using expert knowledge to train the machine-learning model¹⁵⁸. However, this method has limitations. For example, some scenarios will not be solved by experts, and expert knowledge is not accessible to all researchers. Other researchers studied

The only-force sensing is also used in the aerial manipulator trajectory control¹⁵⁹. Like this study, the task was manipulating the aerial robot to scan the flat table surface with “push and slide operations”. Medical UT also used force-guided method¹³⁵. In their study, only force measurement was used to optimise the trajectory during scanning. The RL model was used to build the relationship between measured force and output force which help adjusting the end-effector. After the training, the RL model should output a target torque for the end-effector to rotate the probe. And the RL was trained on real object directly, which is not ideal and will be improved in this study. According to the prior literature, it can be proved that hypothesis of “using only force perception to optimise dynamic trajectory of robotic scanning task” is possible.

RL is an artificial intelligence algorithm that does not need labelled data library like other deep neural network does, but trains the agent like real human does. Human being intakes the examples from real life to ‘train’ him/herself to do certain things. The intake information was not labelled and the actions from the training can be random, and different people may have different actions for the same target. Similar to this routine, RL trains the agent without fixed routine, but give rewards to the better solutions. The most classic example is that the RL trained agent, ‘AlphaGo’ beat the world champion, Lee Sedol, in the game ‘go’. Nowadays, more and more professional go players are training themselves with RL algorithms, according to Chinese professional go players. RL focuses on the optimal solution of Markov decision processes

(MDP), in which the future states of the environment depend on the current action. Robotic control is very related to MDP, since the action robot takes now will affect the future states. As shown in *Figure 25*, MDP has states like S_0, S_1, S_2, \dots , in different states, there is different probabilities to take actions, such as, at S_0 , it may take actions of a_0, a_1 .

$$P_a(s, s') = P_r(s_{t+1} = s' | s_t = s, a_t = a) \quad (1)$$

The probability in the equation shows the probability of the state at current time step t , with the action a , leads the environment to s' in the next time step $t+1$. The chosen actions will lead the environment to the result states. It is similar in the robotic control, as the target waypoints set to the robot, it takes actions to set its moving speed, position and orientation. To finish a task within limited space and time, the current action will affect the choice of future actions and states.

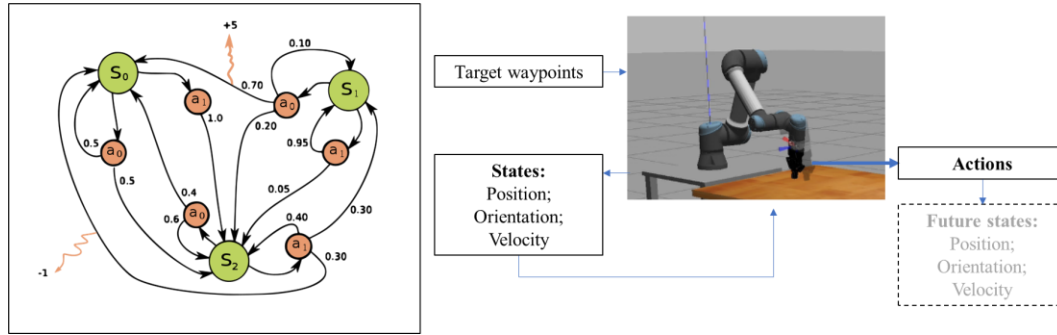


Figure 25 The similarities between MDP in RL and robotic control.

Nowadays, the RL has many algorithms, the main categories are value-gradient and policy-gradient. Since robotic control research is not computer science, the algorithm applied in robotic control must be practical and robust for robot control, but the trend is to use more efficient and effective algorithms. Value-based methods, such as DQN (Deep Q-Network) algorithm, or double DQN, can be used in robotic arm trajectory planning. It can be used to train robot to trajectory in a strange environment¹⁶⁰ and avoid collisions¹⁶¹. However, Value-based methods are typically designed for environments with discrete action spaces, it has been proved that policy-based method has better performance in more dynamic actions and complicated cases, therefore more policy-based methods have been used in research. Such as the REINFORCE algorithm, can be employed to train a robotic system to perform ultrasonic testing efficiently. The policy represents the strategy or behaviour of the robot, and the gradient of the expected reward is used to update the policy parameters. This allows the robot to learn a policy that maximizes the cumulative reward over time. Another policy gradient method-proximal policy optimisation (PPO) was used to optimise the orientation of end-effector in soft skin UT¹³⁵. Active actor critic (A2C) algorithm was used in the surface tracking task of a robotic arm¹⁵¹. In their study, the orientation of the end-effector was output by a neural network,

which is like a look-up table. After the angle recognition, the force control will be implemented using A2C algorithm. Like some prior research, simulation was not mentioned in their study, and the experiments were carried out directly. Tactile exploration using robotic arm was also implemented¹⁵⁰. RL was trained to predict the shape of the object using only tactile exploration on the surface of the object. Soft actor-critic RL algorithm was used in the study. Simulation in Gazebo and experiments in real world were both carried out. But the source code was not posted. A3C is a latest RL algorithm which can be used in a multiple-agent RL task¹⁶². A3C algorithm integrated with distributed learning method, will significantly improve the efficiency of RL, however, it requires more calculation resource and needs more detailed settings.

To implement robotic control on robotic arms, middle ware is needed. Middleware is a type of software or a platform to communicate and coordinate between different systems and components. To integrate the simulation, the control of the simulation model, the control of the real robot, the integration with libraries of algorithms and application programming interface (API) to coding languages^{163 164}. Some kinds of middleware can be used in academic research areas.

Matlab is the most used platform; it has an extension toolbox application of robotic control. It is easy to implement control, however for some types of robots, it is difficult to communicate with the real-world robot. Many researchers use robot operating system (ROS) since it is open-source and feasible for most types of robots. It has a better overall performance and supported by most software. Gazebo is a high-fidelity simulation software based on ROS. Not only simulation software, Gazebo can also carry out the control on the real robot, which makes it outstanding from other current popular software¹⁶⁵. Mujoco, as a popular software in machine learning, has a good physics engine for simulation. However, it is difficult to customise the world environment in it, with lower degree-of-freedom trajectories compared to Gazebo; thus, it is not for real robot control. Considering the flexibility, Gazebo and Webot can work with multiple threads, while Mujoco and Pybullet can work with only one core of the computer processor¹⁶⁵. Since Mujoco is not a free open-source middleware, and it is limited to the complexity of the simulation environment, it was not considered in this study. Unity 3D is another simulation tool that is used by many researchers¹⁶⁶, which was mainly focusing on 3D simulation of animations but can be used in robotic simulation. Morse is similar to Gazebo, but the support and compatibility for robots are worse. CoppeliaSim¹⁶⁷ (formerly V-REP) is a commercial software mainly for industrial robots. It is difficult to modify the world setup of the simulation scene. Blender is a general animation platform; it performs well in robot simulation with a good real-time performance. RoboDK is a simulation tool that draws attention gradually. It has an easy-to-use user interface; it is easy to establish simulation environment and tested in the author's lab practice. However, it also uses OPC UA protocol, which is

difficult to implement communication with a real robot ¹⁶⁴. After free trial test on the computer, it was realised that payment is needed when achieving full functions. Moreover, it is more complex to customise the robot's controller.

To review the research on middleware, the number of publications on “Scopus”, “Web of science” and “ProQuest” databases since 2012, using the keywords “robotic arm”, “simulation” and the name of robot simulation platform and software listed in Figure 26. After comprehensive consideration, ROS was chosen as the middleware of this study since it has functions such as simulation, control of a real robot, and API to coding languages. ROS is open-sourced, so the establishment of the simulation model can be easy. The system will be more stable, and the randomness of customisation of the system and software is better than that of other middlewares.

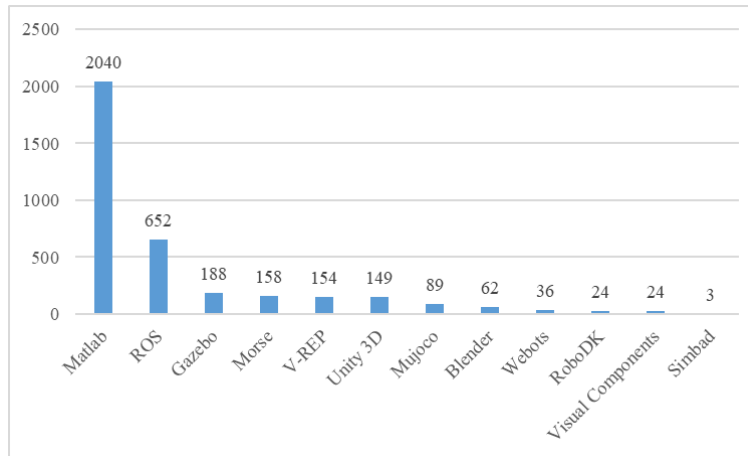


Figure 26 The numbers of research on typical middleware used for robotic research.

To establish a RL environment for robotic arms, many researchers have contributed their methods. Some researchers used Mujoco to build the simulation models ^{168 169}. Openai_Gym with Mujoco was used as a classic suite by RL researchers. However, Mujoco has a disadvantage that it is not open-sourced, so the researchers cannot customise the environment. It does not support inverse kinematics or path planning. With the lower fidelity of Mujoco, it is relatively complex to transfer the results from simulation to reality. Pybullet was also used to fill the gap ¹⁷⁰. Pybullet+Mujoco+Openai_Gym was the solution to implement deep deterministic policy gradient (DDPG) algorithm with hindsight experience replay (HER).

On the other hand, some researchers used ROS to build the simulation model and environment ¹⁷¹. ROS+ Gazebo+ Moveit+ Openai_gym+ Stable_baselines3 was used to build an “MoveRL” simulation platform. Moveit is an embedded solution software which includes classic inverse solvers for robotic arms. This platform was focusing on visual simulation Gazebo+RViz only on safety issues in robotic control with a PPO algorithm. Other researchers used Gazebo only

for controller realisation and Mujoco solution for additional RL implementation ¹⁷².

CoppeliaSim (V-Rep) was used as the simulator ¹⁵², some researchers integrated Matlab RL library on top of V-Rep ¹⁷³, some others directly used python API, such as Tensorflow ¹⁷⁴. CoppeliaSim is a good simulation platform, however, some of the features need license and with limited documentation and community support, it makes CoppeliaSim a not friendly software for beginners. The lack of inverse dynamics makes CoppeliaSim a software relying on plug-ins.

RoboDK was used as the simulation software to integrate python to implement RL algorithms ^{131 173}. However, the studies of RL in RoboDK are most on vision which is because the control of robot is not complete nor open source in RoboDK. Above all, ROS is chosen to be the middleware used in this study. It can be not only used in this study, but also used in future research of our research group for robotic remanufacturing or 3D printing.

2.4 Gap in knowledge and contribution of this study

As reviewed, the prior literature has studied methods to approach robotic UT. However, a knowledge gap exists as human operators can scan an object by only touching it, how to implement that on a robotic arm has not been studied yet. Moreover, previous researchers have not studied the application of robotic UT on dies/moulds. Some algorithms were used in the literature, but the scanning has never been automated.

Less literature considers traditional UT on the surface of manufacturing equipment to be remanufactured. The specialities are that the size and condition of the objects are different from the original, the trajectory of the robotic arm for traditional UT on surface scanning is different from the immersion UT or PAUT, and, more importantly, how to improve the adaptability to implement the UT on different objects without importing the CAD model.

Most of prior literature needs pre-planning of trajectory for the robot, which is not suitable for manufacturing equipment to be remanufactured. The gravity component and bias force component have not been considered during planning. And the optimisation of trajectory between waypoints is not considered. The accuracy of the orientation control can be improved, moreover, the adaptability of these implementations can be optimised. When applying RL on a robotic surface scanning task, the tuning of parameters is complicated and difficult, especially applying the simultaneous control of position and force. In this study, a conventional ultrasonic probe for industrial use will be installed on the end-effector of a 6 DOF robotic arm. A 6 DOF force/torque sensor is installed on the robotic arm to measure the contact force/torque. As real-time force/torque is measured, gravity component and bias force component will be compensated to improve the orientation of the robot. Reinforcement learning algorithm will be

used to train the robot adapt its moving speed and orientation when moving. The algorithm is tested in simulation and experimented in the real world.

Even though the ROS platform is open sourced, the establishment of a simulation model from scratch is not an easy task, especially an advanced compliance controller should be carried out for the orientation optimisation and the RL algorithm should be implemented on the high-fidelity simulation software, Gazebo. Moreover, the synchronisation between the simulation and the real world, and the control of the real robot need additional effort and customised settings for the controllers. In prior literature, there are many papers discussing about orientation optimisation in robotic UT. However, the static orientation optimisation is easier to do, while the optimisation during whole dynamic UT process is difficult. In this thesis, not only the orientation, but also the position, the moving speed are optimised using RL. The efficiency of RL solution is the challenge. Moreover, the transfer from simulation model to the real experiments is challenging, since in the simulation environment, the parameters of the object is known, while in real world, the object is completely unknown and totally different from the object in simulation. Another challenge is that a small single-element UT probe is used in this study, which has never been used in prior robotic UT research.

The contribution of this study will be a new type of robotic arm simulation platform, a new control method of trajectory planning of robotic arm based on RL algorithm, and the realisation of RL real-world experiments. In this study, a robotic arm platform based on ROS will be established to fulfil all the tasks needed for the robotic UT. A simulation model with reinforcement learning algorithm training will be built based on ROS, Gazebo, and Openai_ros package. The reason why Gazebo is chosen, is due to its high-fidelity. "High fidelity" generally refers to the degree of accuracy and realism in the representation of a system or a simulation compared to the real-world counterpart. In various contexts, high fidelity implies a high level of detail, precision, and faithfulness to the original, leading to a more accurate and realistic representation. Because the RL algorithm will be transferred from simulation to the real world, the higher the fidelity the simulation environment has, the easier the transfer will be. Naturally, relatively high computational cost will be paid. The controller will be transferred to the real world to realise the experiments. This study will fill the research gap that the accurate autonomous control of industrial robotic UT using a COBOT and a single-element ultrasonic transducer. The trajectory optimisation algorithm will not only automate the processes of UT, but also improve the critical parameters, such as moving speed during the surface scanning, which is very crucial during scanning on a complex shaped surface of the object. According to the knowledge of the author, this is the first study on the application of robotic UT on the detection of subsurface cracks on machinery tools, i.e., dies and moulds.

A Gazebo + Openai_ros + Stable_baselines3 combination to implement RL algorithms is carried out. It trains the control model to make surface scanning for UT scanning on the unknown surface of machinery tools. The controller is transferred to the real world to implement experiments. The brief overall research strategy can be shown in Figure 27.

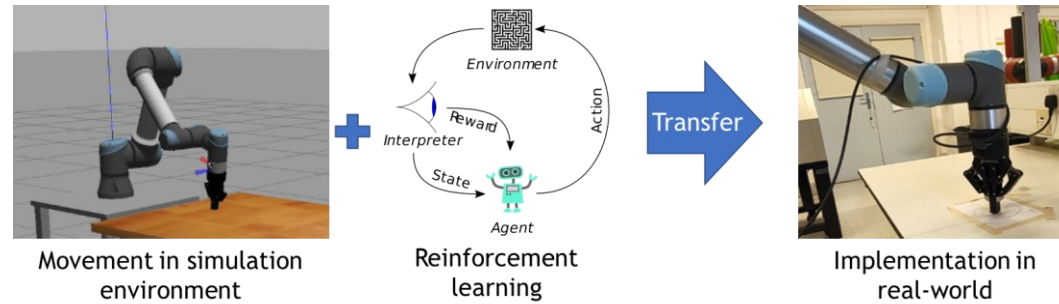


Figure 27 Overall research strategy of this study.

3. The Establishment of Simulation Model and Methodology

To realise a precise robotic control and advanced orientation optimisation using RL algorithm, a reliable simulation environment is necessary. The simulation environment will provide the condition and accessories needed for the proposed controller, evaluate the performance of the robot. The simulation environment will also increase the efficiency of research, since the robot and the environment can be set as planned in simulation but may not be applicable in real world. Simulations also allow researchers to replicate a wide range of scenarios and environmental conditions easily. This capability is valuable for testing the robustness and generalizability of control algorithms across different situations, which may be challenging or time-consuming in the physical world. Regarding accessibility, due to COVID, not everyone has access to specialized robotic hardware during pandemic. Simulation environments democratize access to research in robotic arm control by enabling researchers, regardless of their location or resources, to participate in experiments and contribute to the field. The reason is also that during the RL process, the actions of the agent are random, and it can be dangerous in the real world (both to the robotic arm physical parts and to human operators), simulation is a reasonable option to carry out the training, then the model can be transferred to the real world. In this study, robot operating system (ROS) is selected as the middleware of the computer and the physical robot side.

3.1 Research methodology

The research methodology used in this study is as follows: to improve the reliability of simulation and improve the training speed of the RL training, a robotic simulation model is established first. The robotic model will be introduced in this section. The model will be first studied in the theoretic way. The theoretical model will be helpful when implementing the robot in the simulation platform. The model includes the forward dynamics formula and the reverse kinematics formula. The purpose of these formulas is to calculate the target position of the end-effector and confirm the joint angles according to the location of the end-effector. The theoretic model can be compared with ROS simulation model to make sure the simulation model is correct.

After the literature review, ROS was chosen as the simulation platform. Therefore, a simulation model and surrounding environment should be built in ROS. Most of the studies use ROS to implement tasks to realise the objectives. The ROS platform will be used as the simulation environment and it can be also used to connect the real robotic arm.

In the simulation environment, several simulation experiments will be carried out to achieve the target. At first, the robotic arm model should be built and finish the compliance controller

tests. The compliance controller should be able to control the end-effector to move in a straight line or curved line on a flat surface with constant contact force. Then, RL algorithm should be integrated to the compliance controller. The RL model will be an overall controller harnessing the compliance controller to implement trajectory. A basic “target reaching” test should be carried to verify the RL model is valid. Then, the RL model should be able to trajectory the robot with the compliance controller on a flat surface. To add the difficulty, the RL+ compliance controller should be able to drive the robotic arm on a curved surface. At last, the trained model on the curved surface should be able to transfer to the object. At this stage, the object model is built up in the simulation environment. The trained RL model will optimise the end-effector's trajectory on the object model.

After the optimised trajectory is implemented in simulation. The trajectory will be transferred from the simulation to the real world. The target position of the end-effector will be transferred from the object's location in the simulation to the object's location in the real world. In the real-world application, a probe is attached to the end-effector via a fixed holder. The EPOCH 650 will be connected to the probe to acquire UT signals. After each step of movement, the result from EPOCH can be exported to the PC in .csv format and the results can be visualised. The result is a curve of the amplitude of the UT wave at the probe location. The resulting amplitude of UT can indicate how thick the object is (back wall echo) and how deep the detected defects are.

A small-sized die used in manufacturing is chosen for the object. It has an irregular curved surface. Artificial defects, i.e., drilled flat bottom holes (FBH), are introduced in the die to imitate the internal defects, i.e., the subsurface cracks. The FBHs are located at different locations of the die to verify the capacity of UT. Vertical and horizontal FBHs are both considered in this study. After multiple-time application of the robotic UT, the The probability of detection (POD) of the proposed method can be achieved.

To validate the proposed method, the following experiments are planned: experiment 1 (E1) is to verify the accuracy of the model and repeatability of the model in simulation. E2 is to validate the synchronisation of the simulation model and the real world. E3 is to fine tune the control parameters of the compliance controller for surface scanning. E4 is to test the implementation of RL in ROS simulation with simple tasks. E5 is to compare different RL algorithms on the same scenario. E6 is to tune the hyperparameters of the proposed RL algorithm. E7 is to validate the proposed method in the scanning task on a simple curved surface. E8 is to apply the proposed method on the real object in the simulation. E9 is to measure the real robot's contact force of the compliance controller on a flat surface. E10 is to verify the proposed method on the real object.

3.2 Kinematic and dynamics model

In this study, UR5e is used as the robot. To implement all robot control in simulation and real-world, inverse kinematic (IK) control is needed because it plays a key role in commanding manipulators. The control platform should calculate the robot joint demand according to human's requirements for the end-effector. Complete kinematic and dynamic solutions of robots need to be verified before being employed for modelling and simulation, particularly when they are used to command advanced control states. Therefore, the kinematic and dynamics model of the robot is necessary. They can be used to describe the relationship between the joint angles or positions and the position and orientation (pose) of the end-effector (or any other point of interest) of the robot. The kinematic model is typically used to calculate the forward kinematics (end-effector pose given joint angles) and inverse kinematics (joint angles required to achieve a desired end-effector pose). It is important for motion planning, trajectory generation, and controlling the robot's position and orientation in workspace. The dynamic model describes how forces and torques affect the motion of the robot's joints and the resulting motion of the end-effector. This model takes factors into account such as inertia, friction, gravity, and external forces acting on the robot. The dynamic model is essential for designing and controlling the robot's motion to achieve desired tasks while considering physical constraints and optimizing performance. It is used in simulation, control algorithms, and motion planning to ensure accurate and efficient operation of the robot. Since the target in this study is not only to control the force, but also the position and pose of the robot, both the models are important.

The kinematic model of the Universal Robots UR5 robotic arm describes the relationship between the joint angles or joint positions and the pose (position and orientation) of the end-effector. The UR5 is a 6-degree-of-freedom (DOF) robot, meaning it has six joints that can be controlled to position its end-effector in 3D space.

The kinematic model of robot is often expressed using Denavit-Hartenberg (DH) parameters, which provide a systematic way to represent the geometry and joint connections of a robot. The DH parameters for the UR5 are configured in the URDF (Unified Robot Description Format) files of ROS, which are XML-based files used for describing robot models.

DH parameters are a set of parameters commonly used to describe the kinematic structure of robotic manipulators. These parameters define the geometry and spatial relationships between consecutive robot links and joints in a standardized manner. DH parameters are widely used in robotics for forward and inverse kinematics calculations. The DH parameters consist of four parameters associated with each joint and link of the robot: Link Length (a): The distance between the Z axes of consecutive joints measured along the common normal between the two

joint axes. Link Twist (alpha): The angle between the X axes of consecutive joints measured about the common normal between the two joint axes. Link Offset (d): The distance between the X axes of consecutive joints measured along the previous joint's Z axis. Joint Angle (theta): The angle between the Z axes of consecutive joints measured about the previous joint's Z axis.

These parameters are typically represented as a table, with each row corresponding to a joint and link pair. The table is often organized in a way that facilitates sequential multiplication of transformation matrices for forward kinematics calculations. Here is an example of a simplified kinematic model for the UR5 using DH parameters (see Table 5):

Table 5 DH parameters of UR5e robot.

a(mm)	Alpha(α_i)(rad)	d(mm)	Theta	Mass(kg)
0	pi/2	162.5	θ_1	3.761
425.0	0	0	θ_2	8.058
392.2	0	0	θ_3	2.846
0	pi/2	133.3	θ_4	1.37
0	-pi/2	99.7	θ_5	1.3
0	0	99.6	θ_6	0.365

Here, θ_1 to θ_6 are the joint angles for joints 1 to 6.

Homogeneous Transformation Matrix (Forward Kinematics) can be used to solve direct kinematic (DK) problem. The forward kinematics equation to find the end-effector pose (position and orientation) given the joint angles is obtained by multiplying the individual transformation matrices for each joint:

$${}^0T_6 = {}^0T_1 * {}^1T_2 * {}^2T_3 * {}^3T_4 * {}^4T_5 * {}^5T_6 \quad (2)$$

Where i_jT represents the homogeneous transformation matrix from frame i to frame j. The resulting matrix 0_nT represents the transformation from the robot base frame (frame 0) to the end-effector frame (frame n).

$${}^{i-1}_iT = \begin{bmatrix} c_i & -s_i \cos \alpha_i & s_i \sin \alpha_i & a_i c_i \\ s_i & c_i \cos \alpha_i & -c_i \sin \alpha_i & a_i s_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

The actual URDF file for the UR5 in ROS provides detailed information on the kinematic model, including DH parameters, joint limits, and other necessary information. You can find the URDF

file for the UR5 in the ROS package associated with the UR5 model you are using. The URDF file is typically located in the `ur_description` package.

$$\begin{aligned}
 {}^0T_6 &= {}^0T_1 * {}^1T_2 * {}^2T_3 * {}^3T_4 * {}^4T_5 * {}^5T_6 \\
 &= \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)
 \end{aligned}$$

This is a simplified representation, and for accurate control and motion planning, you might want to use ROS packages like MoveIt! that provide more advanced tools for working with the UR5's kinematic model. After doing this, the coordinate frame for each joint can be transferred from the base frame to the end-effector (see Figure 28). UR5e is a 6 DOF robot with only rotating joints. Frame 0 represents the base link and frames 1 ~ 6 represent the other joints (see Figure 119 in Appendix A).

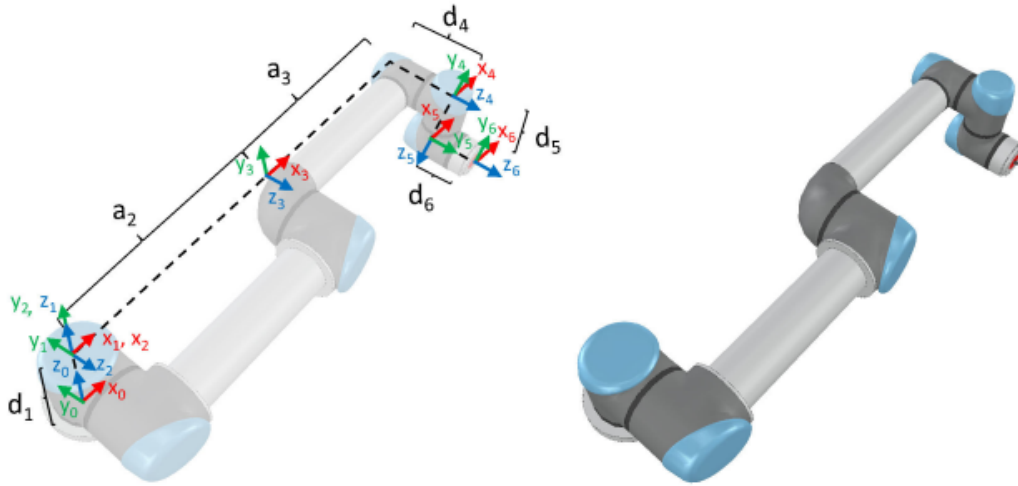


Figure 28 Coordinate system assignment of UR5e.

The homogeneous transformation matrix that uses the classic DH parameters is shown in (2), this matrix represents frame i with respect to frame $i-1$. The position and rotation of the end effector with respect to the robot's base can be calculated by multiplying the 6 transformation matrices as shown in (4). The following equations are used in the (4). θ_i is the same in Table 5 as the joint angle.

$$s_i = \sin\theta_i; c_i = \cos\theta_i \quad (5)$$

$$s_{ij\dots} = \sin(\theta_i + \theta_j + \dots); c_{ij\dots} = \cos(\theta_i + \theta_j + \dots) \quad (6)$$

$$r_{11} = c_1 c_{234} c_5 c_6 + c_6 s_1 s_5 - c_1 s_{234} s_6 \quad (7)$$

$$r_{21} = c_{234}c_5c_6s_1 - c_1c_6s_5 - s_1s_{234}s_6 \quad (8)$$

$$r_{31} = c_5c_6s_{234} + c_{234}s_6 \quad (9)$$

$$r_{12} = -c_1c_{234}c_5s_6 - s_1s_5s_6 - c_1c_6s_{234} \quad (10)$$

$$r_{22} = -c_{234}c_5s_1s_6 + c_1s_5s_6 - c_6s_1s_{234} \quad (11)$$

$$r_{32} = -c_5s_{234}s_6 + c_{234}c_6 \quad (12)$$

$$r_{13} = -c_1c_{234}s_5 + c_5s_1 \quad (13)$$

$$r_{23} = -c_{234}s_1s_5 - c_1c_5 \quad (14)$$

$$r_{33} = -s_{234}s_5 \quad (15)$$

$$p_x = -c_1c_{234}s_5d_6 + c_5s_1d_6 + c_1s_{234}d_5 + s_1d_4 + c_1c_{23}a_3 + c_1c_2a_2 \quad (16)$$

$$p_y = -c_{234}s_1s_5d_6 - c_1c_5d_6 + s_1s_{234}d_5 - c_1d_4 + c_{23}s_1a_3 + c_2s_1a_2 \quad (17)$$

$$p_z = -s_{234}s_5d_6 - c_{234}d_5 + s_{23}a_3 + s_2a_2 + d_1 \quad (18)$$

In this study, a one DOF 2 finger ROBOTIQ gripper is installed on the end-effector of UR5e. This gripper added another configuration to the DH model of the robot. Moreover, the real end-effector and base frame transform matrix needs calibration, which will be described in Chapter 5.

Finger Length: assume the length of each finger as L , 38 mm. when fully closed, the total length of the gripper is 162.8 mm. when holding the transducer holder, the open percent is 72%. the length of the gripper is then, 190 mm, including the transducer holder and transducer.

Finger Position: The position of each finger along its linear path can be represented by a single variable, typically denoted as x_i , where i represents the finger index ($i = 1$ for the first finger and $i = 2$ for the second finger). This variable represents the displacement of each finger from its fully closed position.

End-Effector Pose: Since the gripper has a fixed orientation and does not have rotational DOFs, the pose of the gripper's end-effector can be determined solely by the positions of the two fingers. The end-effector pose is usually represented by the position of the center point between the two fingers. The forward kinematics equation computes the position of the gripper's end-effector based on the positions of the two fingers, i.e., the x_{center} , the position of the gripper's end-effector along the linear path is the average of the positions of the two fingers.

The actual kinematic model for the 2F-85 Robotiq gripper may include additional parameters such as joint limits, maximum opening width, and calibration offsets. These parameters are typically provided by the manufacturer or documented in the gripper's specifications.

To implement control of simulation and real robot, not only DK, but also Inverse Kinematics (IK) is necessary. IK is a mathematical process used in robotics to determine the joint configurations (joint angles or positions) of a robotic manipulator that will result in a desired end-effector pose (position and orientation). In other words, inverse kinematics deals with calculating the joint variables necessary to achieve a specific end-effector position and orientation. The formulas used in inverse kinematics vary depending on the specific kinematic structure of the robotic manipulator. However, the basic idea is to solve a set of equations that relate the joint variables (typically denoted as θ) to the desired end-effector pose (typically denoted as X, Y, Z for position and roll, pitch, yaw for orientation). For the IK, several solutions can be used to solve the problem for UR5e. To solve IK, in the Denavit-Hartenberg (DH) convention, a series of homogeneous transformation matrices are used to represent the transformations between consecutive links of the robot. These transformation matrices are often denoted as A, B, C, D, E , and F , and they represent the transformations from one coordinate frame to another along the robot's kinematic chain.

$$A = p_y - d_6 r_{23} \quad (19)$$

$$B = p_x - d_6 r_{13} \quad (20)$$

$$C = c_1 r_{11} + s_1 r_{21} \quad (21)$$

$$D = c_1 r_{22} - s_1 r_{12} \quad (22)$$

$$E = s_1 r_{11} - c_1 r_{21} \quad (23)$$

$$F = c_5 c_6 \quad (24)$$

$$\theta_1 = \pm \text{atan2} \left(\sqrt{B^2 + (-A)^2 - d_4^2}, d_4 \right) + \text{atan2}(B, -A) \quad (25)$$

$$\theta_5 = \pm \text{atan2} \left(\sqrt{E^2 + D^2}, s_1 r_{13} - c_1 r_{23} \right) \quad (26)$$

$$\theta_6 = \text{atan2} \left(\frac{D}{s_5}, \frac{E}{s_5} \right) \quad (27)$$

$$\theta_{234} = \text{atan2} (r_{31} F - s_6 C, FC + s_6 r_{31}) \quad (28)$$

$$K_C = c_1 p_x + s_1 p_y - s_{234} d_5 + c_{234} s_5 d_6 \quad (29)$$

$$K_S = p_z - d_1 + c_{234} d_5 + s_{234} s_5 d_6 \quad (30)$$

$$c_3 = \frac{K_S^2 + K_C^2 - a_2^2 - a_3^2}{2a_2 a_3} \quad (31)$$

$$s_3 = \sqrt{1 - c_3^2} \quad (32)$$

$$\theta_3 = \pm \text{atan2}(s_3, c_3) \quad (33)$$

$$\theta_2 = \text{atan2}(K_S, K_C) - \text{atan2}(s_3 a_3, c_3 a_3 + a_2) \quad (34)$$

$$\theta_4 = \theta_{234} - \theta_2 - \theta_3 \quad (35)$$

Where atan2 is inverse tangent function, which gives results between -pi to pi.

Experiment 1 (E1): This experiment is to verify the accuracy of the model and repeatability of the model using the inverse kinematic formulas. After building the model, 10 positions in the workspace were set to check the results of the kinematic model. Then, the locations of the end-effector are input to the inverse kinematic formulas. The required joint angles are calculated with the formulas. The robotic arm is driven to each target joint angle with the calculated joint angle. The actual end-effector's location coordinates are compared with the theoretical location coordinates. The average error for the ten locations is 1.02mm (see Table 6). The actual coordinates' locations can be read from the ROS system. ROS system uses forward kinematics to calculate the location of the end-effector. It first acquires joint states from published states and then transforms the joint angles to the location of the end-effector. The differences between target and actual are because of the system error in the ROS robot model and the error in the inverse kinematic calculations. For example, the ROS-based URDF model may introduce systematic errors due to simplifications or parameter inaccuracies in the robot's virtual representation. For the rest of the thesis, there will be more experiments which will be numbered, and the findings in the experiments will be introduced.

For some specific tasks, such as welding, velocity control is used for control. Velocity kinematic is to solve the control parameters of UR5e. Velocity kinematics provides a way to relate the velocities of the robot's joints to the velocities of its end-effector (tool or gripper).

Table 6 Comparison of target and actual positions of end-effector

No	$\theta_1(^{\circ})$	$\theta_2(^{\circ})$	$\theta_3(^{\circ})$	$\theta_4(^{\circ})$	$\theta_5(^{\circ})$	$\theta_6(^{\circ})$	Target	Actual
1	204.05	321.30	251.00	235.03	93.01	32.89	(-0.14042, 0.07769, 0.53325)	(-0.14135, 0.07751, 0.53342)
2	104.10	321.05	234.00	235.12	93.00	33.53	(0.09810, 0.13599, 0.40391)	(0.0996, 0.13628, 0.4045)
3	105.00	-45.20	262.00	235.00	91.00	153.26	(0.09670, 0.14741, 0.60260)	(0.0969, 0.14762, 0.6035)
4	-20.17	-89.09	136.00	-137.02	-89.90	153.85	(-0.39762, 0.00385, 0.20164)	(-0.3978, 0.000371, 0.2012)
5	-256.11	-89.09	-103.57	-62.82	89.37	46.68	(0.01108, 0.51513, 0.43009)	(0.01134, 0.51461, 0.43019)
6	-256.19	-105.02	-100.80	-67.38	90.38	33.68	(-0.00418, 0.57276, 0.29714)	(-0.00423, 0.57279, 0.29588)
7	-254.30	-103.03	-103.63	-66.82	-88.31	11.00	(-0.01495, 0.55657, 0.29509)	(-0.01545, 0.55607, 0.29559)
8	-147.73	-81.95	-125.00	-62.59	-270.09	-238.23	(-0.4015, -0.09573, 0.30672)	(-0.4006, -0.09618, 0.30632)
9	-147.73	-81.50	-124.58	-69.9	89.74	134.58	(-0.39022, -0.08825, 0.29462)	(-0.3894, -0.08874, 0.29397)
10	-6.47	-103.44	-87.42	-74.28	89.81	91.13	(0.57291, -0.19950, 0.41116)	(0.57401, -0.19991, 0.41155)

This relationship allows for real-time control of the arm's motion based on desired end-effector velocities. By understanding how joint velocities translate to end-effector velocities, velocity kinematics enables the generation of smooth and accurate trajectories for the robotic arm. This is important for tasks such as pick-and-place operations, painting, welding, or any application where precise and continuous motion is required.

To solve the velocity of the end-effector, Jacobian matrix is needed to solve each joint according to the end-effector. For a robotic manipulator with 6 degrees of freedom (DOF) 6 joint robot UR5e, the Jacobian matrix J is a 6×6 matrix.

$$\dot{x}_{end-effector} = J(q)\dot{q} \quad (36)$$

Where $\dot{x}_{end-effector}$ is the velocity of end-effector. $J(q)$ is Jacobian matrix, $\dot{q} = [\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5, \dot{\theta}_6]^T$ is the velocity of joint.

The Jacobian matrix is as follows:

$$J = \begin{bmatrix} J_{L_i} \\ J_{A_i} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{i-1} \times \mathbf{r}_{i-1,e} \\ \mathbf{b}_{i-1} \end{bmatrix} \quad (37)$$

where \mathbf{b}_{i-1} is the unit vector representing the z-axis of joint i-1 with respect to the base (frame 0), $\mathbf{r}_{i-1,e}$ is the end-effector position with respect to frame i-1, and J_{L_i} and J_{A_i} represent the parts of the Jacobian that relate the joint velocities to the linear and angular ones.

$$J_A = \begin{bmatrix} 0 & s_1 & s_1 & s_1 & c_1 s_{234} & r_{13} \\ 0 & -c_1 & -c_1 & -c_1 & s_1 s_{234} & r_{23} \\ 1 & 0 & 0 & 0 & -s_{234} & r_{33} \end{bmatrix} \quad (38)$$

Where r_{13} r_{23} and r_{33} are the elements of the rotation matrix R associated with the orientation of the end-effector frame with respect to the base frame.

$$\mathbf{r}_{0,e} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \quad (39)$$

$$\mathbf{r}_{1,e} = \begin{bmatrix} p_x \\ p_y \\ p_z - d_1 \end{bmatrix} \quad (40)$$

$$\mathbf{r}_{2,e} = \begin{bmatrix} p_x - c_1 c_2 a_2 \\ p_y - c_2 s_1 a_2 \\ p_z - s_2 a_2 - d_1 \end{bmatrix} \quad (41)$$

$$\mathbf{r}_{3,e} = \begin{bmatrix} p_x - c_1 c_{23} a_3 - c_1 c_2 a_2 \\ p_y - c_{23} s_1 a_3 - c_2 s_1 a_2 \\ p_z - s_{23} a_2 - s_2 a_2 - d_1 \end{bmatrix} \quad (42)$$

$$\mathbf{r}_{4,e} = \begin{bmatrix} r_{23} d_6 + c_1 s_{234} d_5 \\ r_{23} d_6 + s_1 s_{234} d_5 \\ r_{23} d_6 - c_{234} d_5 \end{bmatrix} \quad (43)$$

$$\mathbf{r}_{5,e} = \begin{bmatrix} r_{13} d_6 \\ r_{23} d_6 \\ r_{33} d_6 \end{bmatrix} \quad (44)$$

Where $\mathbf{r}_{i,e}$ is the rotation transform matrix between i-th frame and end-effector. Related equations can be found from (5) ~ (18).

$$J_{L_1} = \begin{bmatrix} -p_y \\ p_x \\ 0 \end{bmatrix} \quad (45)$$

$$J_{L_2} = \begin{bmatrix} -c_1(p_z - d_1) \\ -s_1(p_z - d_1) \\ s_1p_y + c_1p_x \end{bmatrix} \quad (46)$$

$$J_{L_3} = \begin{bmatrix} c_1(s_{234}s_5d_6 + c_{234}d_5 - s_{23}a_3) \\ s_1(s_{234}s_5d_6 + c_{234}d_5 - s_{23}a_3) \\ -c_{234}s_5d_6 + s_{234}d_5 + c_{23}a_3 \end{bmatrix} \quad (47)$$

$$J_{L_4} = \begin{bmatrix} c_1(s_{234}s_5d_6 + c_{234}d_5) \\ s_1(s_{234}s_5d_6 + c_{234}d_5) \\ -c_{234}s_5d_6 + s_{234}d_5 \end{bmatrix} \quad (48)$$

$$J_{L_5} = \begin{bmatrix} -d_6(s_1s_5 + c_1c_{234}c_5) \\ d_6(c_1s_5 - c_{234}c_5s_1) \\ -c_5s_{234}d_6 \end{bmatrix} \quad (49)$$

$$J_{L_6} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (50)$$

The dynamic equations of a robotic arm describe how the joint torques or forces affect the motion of the arm. These equations take into account factors such as inertia, gravity, Coriolis and centrifugal forces, friction, and external forces acting on the arm. The dynamic model of UR5 is

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = u \quad (51)$$

where $M(q)$ is symmetric positive definite mass inertia matrix of the system, $C(q, \dot{q})$ is the matrix of Coriolis and centrifugal terms. $g(q)$ is the vector of gravity terms and u is the input vector. u is the joints torque vector.

The inverse dynamic equation is:

$$\ddot{q} = M^{-1}(q)(u - C(q, \dot{q})\dot{q} - g(q)) \quad (52)$$

$$M(q) = \sum_{i=1}^n (m_i J_{L_i}^T J_{L_i} + J_{A_i}^T R_i I_i R_i^T J_{A_i}) \quad (53)$$

where m_i is the mass of link i . R_i is the rotation matrix representing the orientation of link i with respect to the world frame. I_i is the inertia tensor of link i with respect to its center of mass.

Singularity is an extreme situation for robotic arms in which they lose some of their degree of

freedom. Singularities were also encountered during the robotic arm's commissioning. The robot fully extended and cannot move to the required position. A limited drive space was defined to prevent workspace singularities so that the robotic arm only moves to the necessary locations for the scanning tasks and would not move to singularity spaces. The singularity space is the edge of the arm's reach. The singularity can also be prevented in the kinematic formulas

175

The Jacobian matrix establishes the relationship between the joint velocities and the linear and angular velocities of the end effector. The singularities happen at the points where the Jacobian matrix becomes rank-deficient ($\det(J) = 0$), then the robot loses the ability to control certain directions of motion. According to the formulas above, $\det(J)$ can be listed as:

$$\det(J) = s_3 s_5 a_2 a_3 (c_2 a_2 + c_{23} a_3 + s_{234} d_5) \quad (54)$$

Three singularities exist for the shoulder, elbow, and wrist of UR5. A shoulder singularity happens when the last factor in Equation (46), which involves angles θ_2 , θ_3 , and θ_4 , is equal to zero. The end effector cannot be moved along z_6 . An elbow singularity is present when $s_3 = 0$, which happens when $\theta_3 = 0$ or $\theta_3 = \pi$. This means that the arm is fully stretched or bent; however, only the former case is physically possible. Wrist singularities exist when $s_5 = 0$, which mathematically happens when $\theta_5 = 0$ or $\theta_5 = \pi$. This renders z_4 and z_6 parallel (see Figure 28). The joint angles above should be avoided. The combined method including the limited use space and joint angles is used in this study to prevent singularities.

3.3 Simulation model of robotic arm UR5e

ROS is a robotic middleware framework that connects the physical and application layers of the robotic arm by enabling communication and data exchange across these layers (introduced in Appendix A: Implementation part). Gazebo is a simulation software based on ROS. A physical environment can be established in Gazebo, such as a room, a table, and all the objects used in this study. Within the physical environment, the robotic arm (see Figure 29) can interact with other objects, such as collisions between objects and computer vision among objects. Within the simulation software, the developers can see the environment of the robot and the movement of the robot. Gazebo has a large library of physical objects, such as van vehicles and cola cans, for the developer to insert into the simulation environment. Apart from that, developers can also render their own designs of objects into the world. Gazebo is not only for simulation visualisation, but also supports realistic physics, sensor simulations. However, within Gazebo itself, it is not interactive. The developers can move the robot using coding API, such as Python, C++, and other planning tool software, such as, RViz. The developers can define the environment with *.world* file, spawn the controller, spawn the robot urdf at specified

location. After the launch of the `.launch` file the Gazebo environment can be started. ‘gazebo_ros_pkgs’: This is a collection of ROS packages that provide Gazebo plugins, models, and other resources for the integration of Gazebo with ROS. It includes packages like `gazebo_ros_control` for connecting ROS controllers to simulated robots.

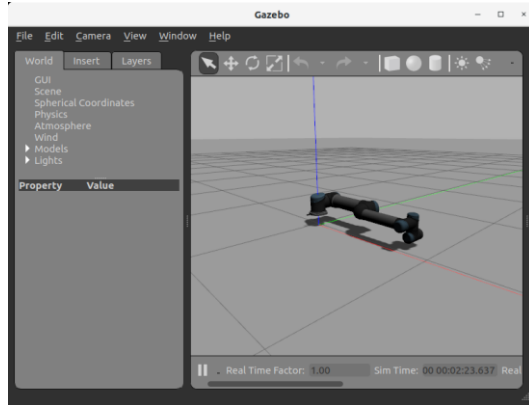


Figure 29 Gazebo simulation of UR5e.

RViz is a “ROS Visualization” software tools, which can be synchronised with Gazebo environment. the defined positions in Moveit of the robot can be directly assigned to the robot in RViz. The `joint_state` controller under `ros_control` can read the states of joints, this controller will be used by RViz to synchronise the robot in RViz and Gazebo. There are many features that can be used in the simulation, such as, “interactive marker” which can be directly dragged and release to define the targeted position of the end-effector. RViz is not only a visualisation tool, but also a planner. There is an open motion planning library (OMPL) embedded it to plan the trajectory of the robotic arm. After the trajectory is planned, the movement of the joints and links can be assigned by Moveit. Algorithms, such as rapid random tree (RRT), can be used to plan the trajectory (as shown in Figure 30).

Besides `roslaunch` file, ROS can also run Python code by using `roslaunch` command, which can directly find the related packages and execute the executable documents under the package. This command is suitable for external controller or command coded by Python or C++. `roslaunch` can only run one node from one package at a time in one terminal window, which is less powerful than `roslaunch` ¹⁷⁶. The novelty of this study is that all the joints of UR5e and the gripper can be controlled not only with Python code, but also with RViz. The advantages of using RViz are that potential trajectory of robot and trajectory parameters can be visualised and extra plug-in tools such as computer vision tools can be used in RViz.

Above all, the robotic control platform based on ROS has been developed in the author’s prior conference article ¹⁷⁷. It is implemented in a Linux operating system. It can finish simulation work, the synchronisation between simulation and real-world. Additional works, such as,

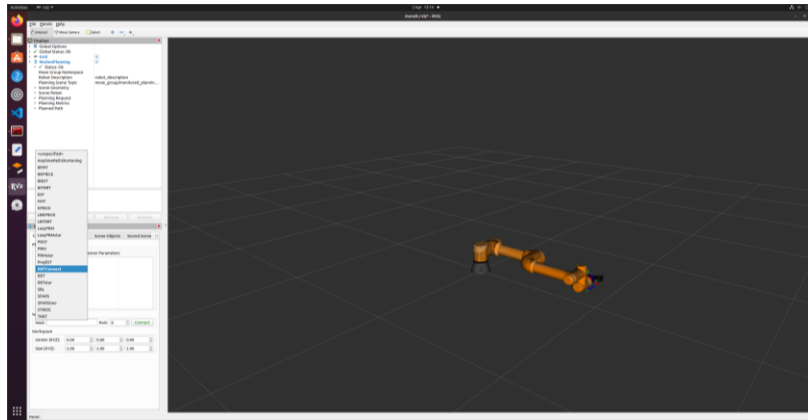


Figure 30 Planners in RViz.

adding gripper, digital twin, computer vision, deep neural network training, can be implemented on this platform.

As shown in Figure 119 in Appendix A, the joints and links of UR5e are set in the .urdf document. The urdf and mesh file are from the open-sourced repository ¹⁷⁸. Based on these packages, the move group and planner can be designed on Moveit software. To make it a complete setup, a 2F-85 gripper from ROBOTIQ company was also installed on the end-effector of the robot. “2F-85” is the type, which means it has two fingers and the maximum distance between the fingers is 85 mm. The urdf file of the gripper can be imported from the repository ¹⁷⁹. An overall urdf.xacro should be established for the “robotic arm+gripper”. Within the urdf.xacro, the base link of the gripper should be fixed to the end-effector of the main robot. To make the gripper move properly like the real-world one, the setup in Moveit is necessary. The processes of Moveit setting are shown in Figure 31. To make the gripper move, a fixed joint must be defined, then, the rotary joint in the gripper must be defined to make it work properly. In the Moveit software, there are some pre-defined positions that can be set. In this study, open (0%) and close (80%) position was set up. After the setup of the gripper, the base link of the gripper is fixed to the end-effector link of the robotic arm. Subsequently, the complete robotic arm is finished in simulation (as shown in Figure 32).

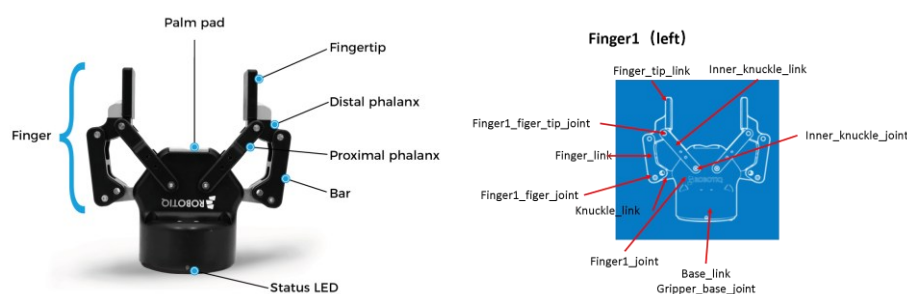


Figure 31 Joints and links in Robotiq gripper (left), and the Moveit setup (right).

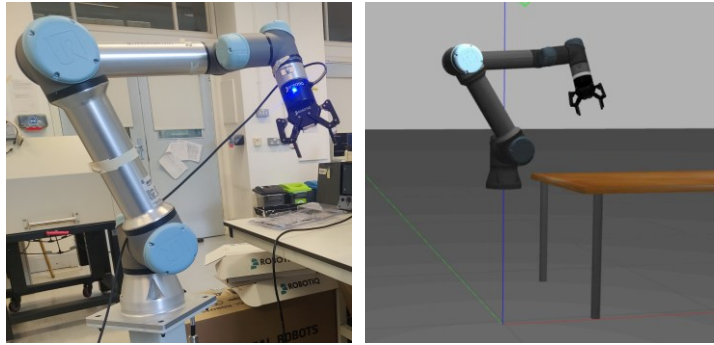


Figure 32 Picture of the real robot and the simulation model in Gazebo.

Another point to make the gripper work properly is to make it “collision-aware” and “able to grip”. These functions are easy in real-world, but difficult to realise in the simulation environment. The processes are explained step-by-step.

To make the gripper collision aware, the setup in the urdf is important. The first thing is to set the inertial and physics parameters properly (see Figure 126 Appendix). In the left figure, the object (a table in this case) SDF file is set to be static. In the surface link, a collision is defined with a name. Within the collision definition, parameters, such as friction coefficient, should be set up. In the right figure, the URDF of the gripper should be defined carefully. In the section of the finger-tip link, a collision is defined. Referencing the name of the collision, a collision contact sensor is defined with the ‘libgazebo_ros_bumper’ plug-in. The detection rate is defined as 15 Hz. A rostopic is also defined to publish the contact position and the contact force. The plug-in will publish a rostopic when the robot collides with a static object, which is built as the .SDF file in the simulation world. After the setup of the collision features, when the robot collides with the static object in the simulation, the rostopic will publish the position and the type of collision in the real-time (as shown in Figure 33). This function is not only for the safety issue in the simulation, but also the preparation for the surface scanning task in the future operation. As the gripper touches the surface of the object, the collision should be notified.

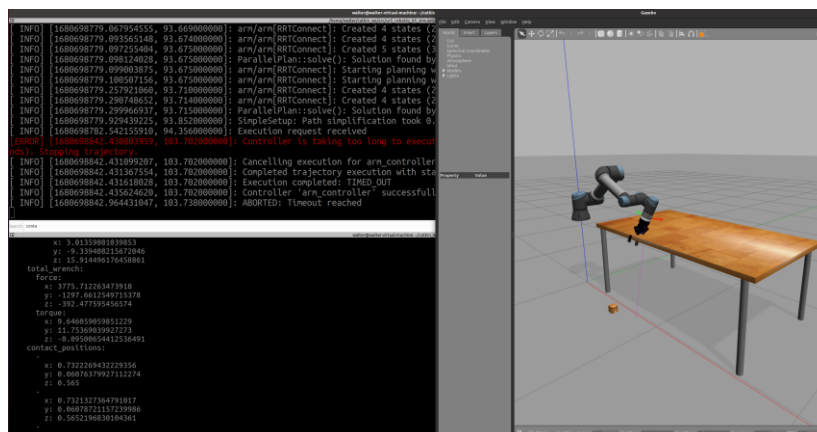


Figure 33 Screenshot of collision happened in Gazebo and the collision detection.

For the gripping task, this task is more for the function test of the simulation model. In previous simulation, the gripper just slipped away from the object, and the object, such as the cube just flew away from the original position. After checking, the problem is the version of Gazebo and the necessity of using a plug-in named “gazebo_grasp_fix”¹⁸⁰. With this gripper setup, the movements of the gripper can be optimised. The details are: when the finger tips of the gripper contact with the object, the PID force controller will take actions, to make the gripping action more realistic. To make sure gripping action more properly finished, these plug-in should be implemented (as shown in Figure 36). Moreover, the hardware interface of the gripper is set to be the type of “EffortJointInterface”, which control the robot basically depending on the effort the robot is under, but not the position, i.e., the conventional interface type “PositionJointInterface”. To change from position controller to effort controller, the transmission of the urdf file should be modified. The most important thing is to change the setup in the ros_controller.yaml, from position_controller to effort_controller.

The feature of UR5e is an embedded force-torque (FT) sensor, which can measure 6DOF force and torque. This sensor is installed in the real robot, but not embedded in simulation model. Therefore, a plug-in for force and torque measurement should be inserted in the ROS package (see Figure 34). To add the FT sensor plug-in, the necessary Gazebo ROS packages should be installed, such as, ‘gazebo_ros_pkgs’ and ‘gazebo_ros_control’. Edit the URDF file of the UR5 to include a force/torque sensor. This involves adding a sensor element to the robot's URDF model to a specific link, wrist_3_link, in this case. Write a Gazebo plugin that simulates the force/torque sensor. This plugin “libgazebo_ros_ft_sensor.so” will generate simulated force/torque readings based on the robot's interaction with the environment. Inside the plugin, you will need to subscribe to the physics update and apply force/torque values accordingly. For simulation, these values will be calculated based on the robot's interactions. After this, a catkin_make is necessary to make it work.

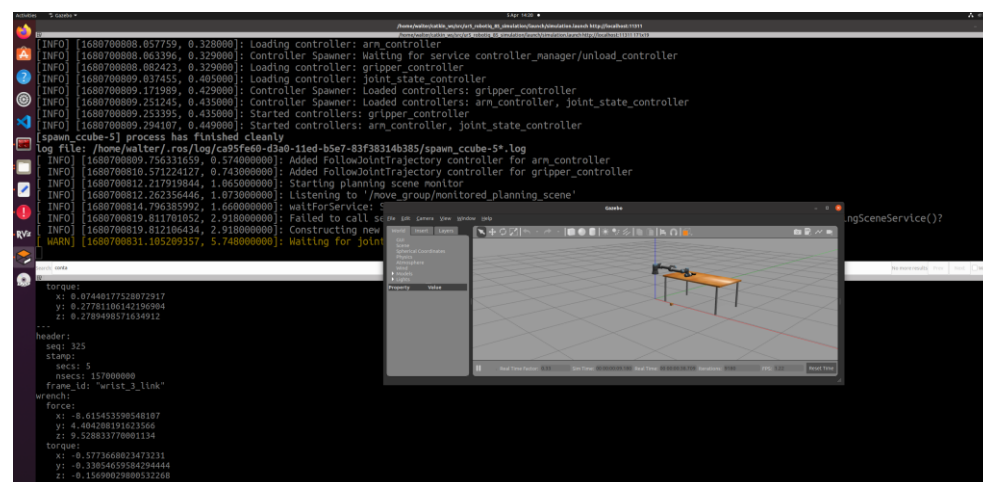
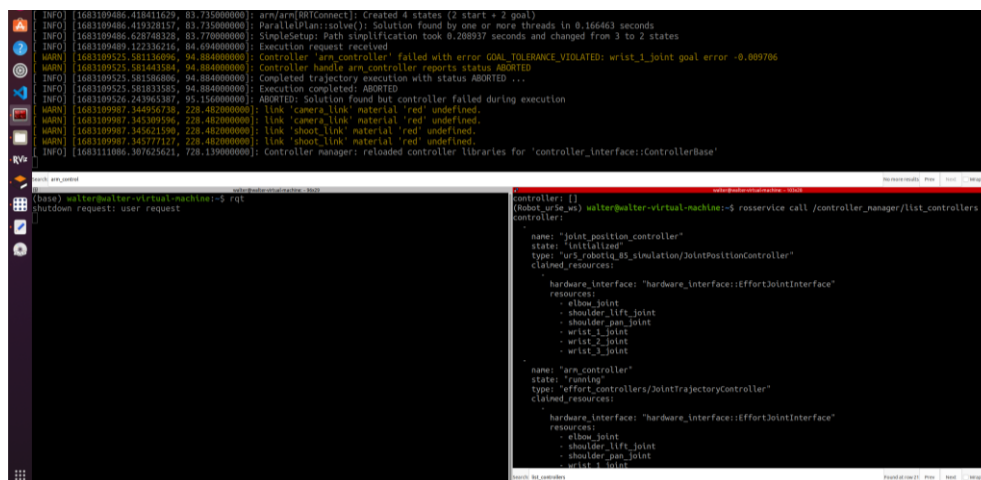


Figure 34 Screenshot of force/torque sensor plug-in in Gazebo simulation.

The plugin establishes a connection with the Gazebo physics engine. It subscribes to updates from the Gazebo simulation loop to retrieve information about the simulated robot and environment. The plugin publishes force and torque readings as ROS messages on specified topics, “wrench” in this case. These topics can be configured based on the plugin parameters or settings in the URDF file. The simulated force-torque sensor data can be used by other ROS nodes, such as controllers or perception modules, to simulate the response of a real-world force-torque sensor. This enables the testing and development of robot control algorithms and perception systems in a simulated environment.

For the configuration of the robot, the controllers of each joint can be defined in the config.yaml (see Figure 127 Appendix). Effort controller is used instead of “position controller” in the control of robot see Figure 35. Since in Gazebo software, the control and physics parameters are complicated to tune. To make the robot actions more realistic, for example, applying forces and pick-and-place, the optimal way is to use effort controller. Effort controller is a kind of controller that controls the torque or effort on the joint of robot. Unlike position controllers, the effort controller does not strictly follow the positions of target. The torque/effort applied to the joint is important during the tasks such as, surface scanning on flat or curved surfaces, pick-and-place task. In Gazebo, if the joints are not set properly, the performance of the particular task, such as pick-and-place (see Figure 36), will be abnormal, i.e., the object jumped from the gripper, or abnormal movement of the links of the gripper. The setup of the effort controllers of the robot is shown in Figure 40. In effort controller package, the controller receives the state of the joints firstly. The information of the position, velocity and acceleration of the joint will be collected. The torque topic will be calculated using a PID controller based on the deviation between current position and the target positions. Once the torque is calculated, the value will be published to the target torque topic. Subsequently, the joint state publisher will update the information of the joint. The processes will repeat for the next time step.



```

INFO [1681109406.418411229, 83.7150000000]: arm_controller: Created 4 states (2 start + 2 goal)
INFO [1681109406.419328157, 83.7150000000]: ParallelPlan::solve(): Solution found by one or more threads in 0.166463 seconds
INFO [1681109406.628748326, 83.7700000000]: SimpleSetup: Path simplification took 0.208937 seconds and changed from 3 to 2 states
INFO [1681109409.122136216, 84.6940000000]: Execution request received
WARN [1681109525.581136096, 94.8840000000]: Controller 'arm_controller' failed with error GOAL_TOLERANCE_VIOLATED: wrist_1_joint goal error -0.889786
WARN [1681109525.581443154, 94.8840000000]: Controller handle arm_controller reports status ABORTED
INFO [1681109525.581868086, 94.8840000000]: Completed trajectory execution with status ABORTED ...
INFO [1681109525.58183585, 94.8840000000]: Execution completed: ABORTED
INFO [1681109526.243965267, 95.1500000000]: ABORTED: Solution found but controller failed during execution
WARN [1681109987.344956738, 228.4820000000]: Link 'camera_link' material 'red' undefined.
WARN [1681109987.345309596, 228.4820000000]: Link 'camera_link' material 'red' undefined.
WARN [1681109987.345621156, 228.4820000000]: Link 'shoot_link' material 'red' undefined.
WARN [1681109987.345777127, 228.4820000000]: Link 'shoot_link' material 'red' undefined.
INFO [168111006.307625621, 728.1390000000]: Controller Manager: reloaded controller libraries for 'controller_interface:ControllerBase'

(based) muller@walter-virtual-machine:~$ rqt
Shutdown request: user request

controller: {}
Robot state set() muller@walter-virtual-machine:~$ rosservice call /controller_manager/list_controllers
controller:
  name: "joint_position_controller"
  state: "initialized"
  type: "ursimbot2_robot_simulation/JointPositionController"
  claimed_resources:
    hardware_interface: "hardware_interface:EffortJointInterface"
    resources:
      - elbow_joint
      - shoulder_lift_joint
      - shoulder_pan_joint
      - wrist_1_joint
      - wrist_2_joint
      - wrist_3_joint
  name: "arm_controller"
  state: "running"
  type: "effort_controllers/JointTrajectoryController"
  claimed_resources:
    hardware_interface: "hardware_interface:EffortJointInterface"
    resources:
      - elbow_joint
      - shoulder_lift_joint
      - shoulder_pan_joint
      - wrist_1_joint

```

Figure 35 Screenshot of the effort controllers used in this study.

Using these settings, relatively complex tasks, such as, pick-and-place, computer vision (see Figure 37) recognition tasks can be realised.

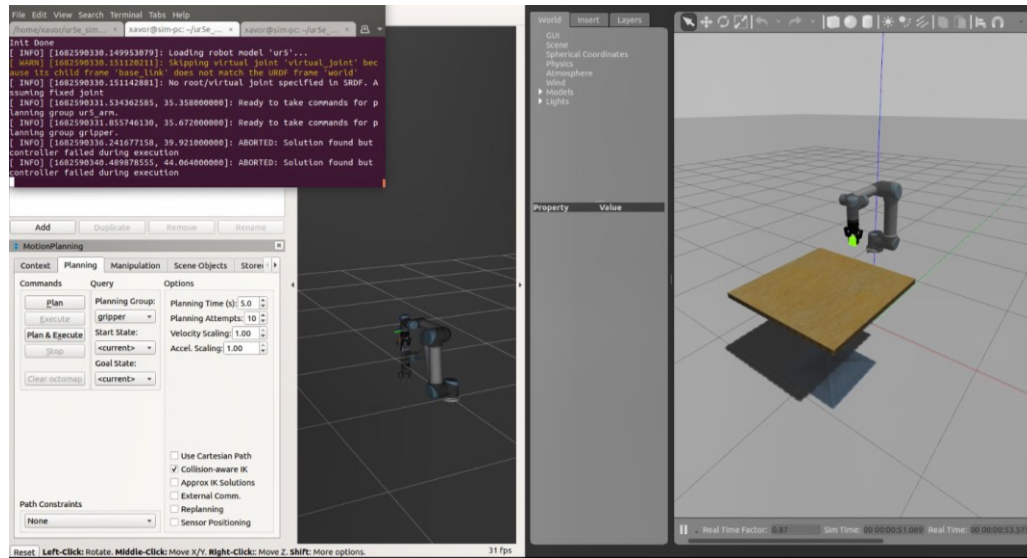
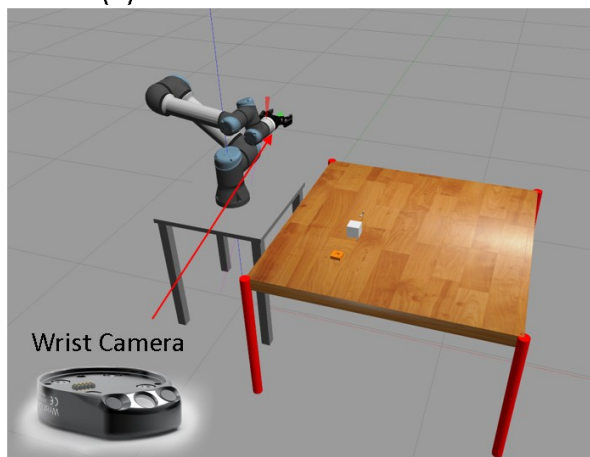


Figure 36 Screenshot of the pick-and-place task.

(a) Gazebo simulation software



(b) Monocular images from different views

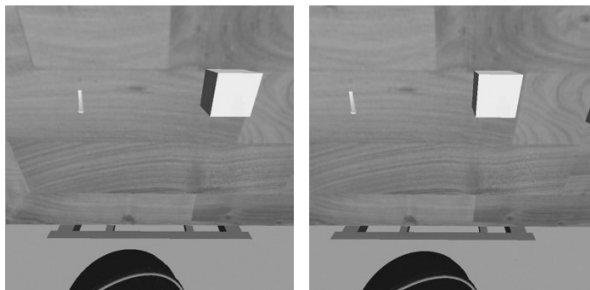


Figure 37 Screenshot of the computer vision task.

To simulate the real-world scenario, the transducer holder was added to the end-effector of the simulation model (see Figure 38). Since the ultrasonic transducer is to be installed on the robotic

arm, a holder will be installed on the robot. It is essential to put the holder on the simulation model. In this way, the simulation model and real robot are more alike, and after the physical environment is set, it is a digital twin model of the robotic UT platform. To add the holder on to the robot, the 3D model of the holder has be fixed as a link on to the gripper link, finger_1 link, in this case. The code should be modified in urdf.xacro file. After that, the configuration of Moveit should be modified accordingly. The details of the transducer holder will be introduced in Chapter 5. To add this holder, the .stl model should be included in the urdf of the robot. Different parts of the holder will be joined together via fixed joint in the urdf. After this, the joints and links information will be updated in Moveit software, then the related moveit_config package will be updated. Within the srdf document included in the planning_context.launch file, the collision information will be updated, otherwise the collision between the holder parts will stop the execution of the simulation.



Figure 38 Simulation model of robotic arm and transducer holder.

To realise the surface scanning task with the simulation model for UR5e, the special place is to insert a force/torque sensor at the last link of the robot, which will close the loop of force/position control of the end-effector. The plug-in code will be added into the urdf document of the robot. The location of the sensor on the link and the name of the link are defined in the urdf file. After the launching of the simulation model, the force/torque will be calculated based on the motion of the robot. The force/torque data is published on the topic “wrench/force” and “wrench/torque” in this study. It is a beneficial method to make the simulation closer to the real world. According to the real situation of real-world, the simulation environment of UR5e can be shown as in Figure 39. With the implementation of the sensor, the model has included all three contact task features, compliance control, tactile sensor and force/position control. With the structure of the controller, the scanning task will be planned with a higher-level controller. This is the innovation of this study.

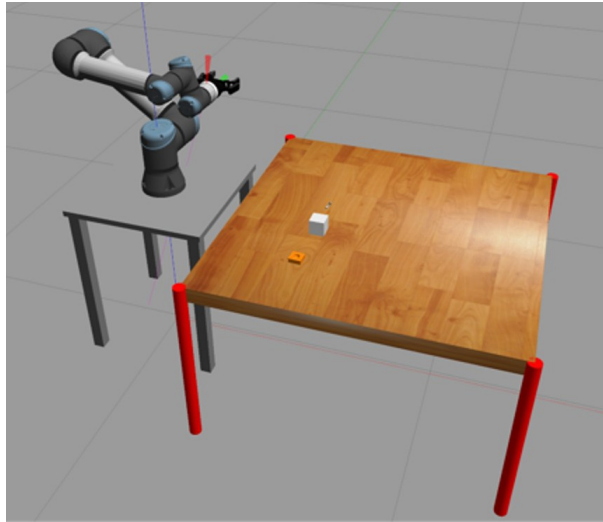


Figure 39 Screenshot of the environment in Gazebo simulation.

To realise surface scanning and other applications in the simulation environment is not as easy as that. Even though it seems that the environment has been set up perfectly, the details of the robot and the objects need further setups, for example, the original pose of the robotic arm, the physics parameters of the object and the world.

After the force/torque sensor has been plugged in ROS, the force/torque sensor can measure the contact forces and torques in three directions of the closed end-effector (shown in Figure 40). It shows the 6DOF force/torque change before and after the contact of the gripper with the table surface. During this process, the robot was only controlled to approach the surface of table. As can be seen the z-axis contact force changed from 0 to -5N, which means it has contact with another surface. Forces on x and y-axis have vibration but not very much. The torque around y-axis had a big change since the end-effector had an orientation error along y-axis, but it is still along with x-axis, so the torque around x-axis stayed stable.

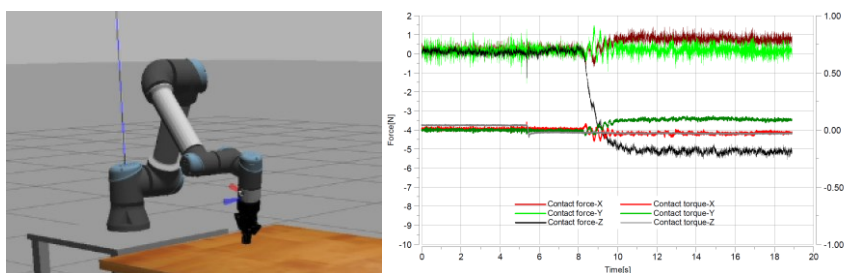


Figure 40 Force and torque measurement in Gazebo simulation.

After the end-effector touched the table surface, the contact force reached -5N. During the touching period, the standard deviation (SD) of force on the z-axis is 0.1395, which means the force on the z-axis is very stable. However, scanning on the surface was not considered in this case. The force on the y-axis has the most significant SD, 0.2398, which can also be seen in the

figure, the force on the y-axis has a big oscillation in the range of $\pm 1.5\text{N}$. This indicates the pose when it touches the surface along the y-axis.

To implement the final task, i.e., surface scanning on an unknown curved surface, some demos are to be created as a pre-trained task, e.g., surface scanning on a flat surface and pick and grasp task. In these tasks, the performance of the effort controller in robotic arm and gripper can be tested. The actual results of these tasks are shown in the following Figure 41: a target -50N constant contact force is set, and the robotic arm scans the flat desktop surface with the transducer holder on the tip of the end-effector. The error is over $\pm 25\text{N}$. The standard deviation is 5.8595, which is big and needs improvement in this study.

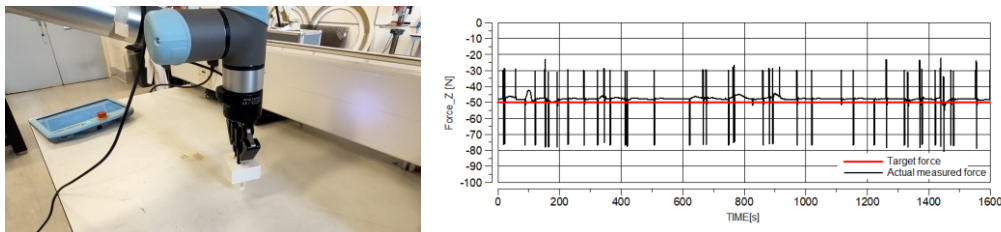


Figure 41 Constant force control on flat surface scanning in real robot environment.

To realise these tasks and the final tasks, more complicated controllers should be implemented in ROS to realise the position/force control. The controllers used in this project are introduced in the next sub-section.

3.4 Controllers used in simulation

In this study, several controllers should be used to realise the proposed function (as shown in Figure 42). The control of the robot switches between different controllers to realise a sequence of tasks in the surface scanning job. The force controller is mainly for the driving force of the end-effector, it will drive the end-effector to approach the target surface and keep the contact force between end-effector and the contact surface in the assigned direction constant. When force controller is in charge, the motion will be driven by target force defined in the rostopic and the motion will be solved by the configurations in dynamic config. The motion of the robot cannot be modified separately since it is not the purpose of force controller. To modify the pose of the robot, the trajectory controller should be used. This controller can not only adjust the position of the robot, but also the orientation of the end-effector. So, the trajectory controller in this task will control the pose movements to fulfil the task, i.e., the orientation adjustment on the target position, the movement between the original home position and target positions. The orientation adjustment is especially important for this task, since the transducer should be normal to the contact surface. With the help of Moveit in ROS environment, the orientation controller will optimise the orientation of the end-effector according to the measured forces to

keep it normal to the contact surface, during the trajectory.

Since the control theories of the controllers above are different, force controller includes the desired force in loop to solve the target pose using dynamics config of the robot, while trajectory controller uses a PD controller to interpolate between target positions and actual positions. Normally, they can only control the robot separately, i.e., it is not possible to control the robot with any combination of two controllers, however, since the compliance controller can cooperate with trajectory controller, it is possible to move the end-effector while applying compliance control ¹⁸¹. The trajectory controller under compliance controller can control the moving speed, the targeted position. On the other hand, this allows us to control the end-effector to implement surface scanning, i.e., with contact force control and position/pose control. If the control on robot can be implemented both in simulation and reality, then the training of the RL agent will be easier. We can only consider the implementation of RL code, but no need to think about implementing the control during RL or in the real world again.

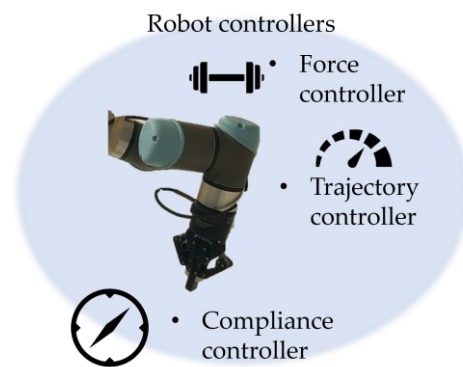


Figure 42 Controllers of the robot used in this study.

Force controller: in ROS there are some kinds of controllers, such as position controller, velocity controller and effort controller. Since force is concerned in this study, effort controller is used as main controller in the configuration of robotic arm. In the configuration of robot, there is a parameter named “max_effort” for each joint. By increasing the max_effort value, it is allowed that the joint exerts a larger maximum force or torque. This can be beneficial when the joint needs to resist external disturbances or apply stronger forces to manipulate objects or maintain stability, for example, to maintain the orientation of the end-effector during scanning on the surface. However, it's important to ensure that the increased effort is within the safe operating limits of the robot and its components. The force controller will take whatever it takes to fulfil the “target_wrench”, therefore it cannot maintain the pose while meeting the force target. The iterations parameters of the force adjustment which defines how many times the solver will iterate the force solution to approach target value in one control cycle are not functional in force controller.

PID Parameters of controllers for each joint are also important (see Figure 43). The PID (Proportional-Integral-Derivative) control parameters, including the proportional gain (K_p), integral gain (K_i), and derivative gain (K_d), directly affect the joint's response to errors, velocities, and accelerations. By tuning these parameters, you can adjust the joint's control behavior to achieve better tracking, stability, and force/torque control. Proportional Gain (K_p): A higher K_p value increases the joint's response to positional errors, allowing it to exert a stronger corrective force. However, a too high K_p value can lead to instability or oscillations. Integral Gain (K_i): The K_i term helps address steady-state errors by accumulating the integral of the error. Tuning the K_i value can improve the joint's ability to maintain position against external forces. Derivative Gain (K_d): The K_d term provides damping and reduces overshoot or oscillations caused by rapid changes in position or external disturbances. Adjusting the K_d value can enhance the joint's stability and response to sudden forces.

```
joint_limits:
  shoulder_pan:
    # acceleration limits are not publicly available
    has_acceleration_limits: false
    has_effort_limits: true
    has_position_limits: true
    has_velocity_limits: true
    max_effort: 150.0
    max_position: !degrees 360.0
    max_velocity: !degrees 180.0
    min_position: !degrees -360.0

gains:
  shoulder_pan_joint:
    p: 1500
    d: 50
    i: 10
    i_clamp: 100
```

Figure 43 Max_effort and PID setup for effort controller.

Motion controller: The cartesian motion controller is used with the help of trajectory planning controller to control the movement in this case. Effort controller and position controller have been tested. An effort controller is also used to control the trajectory when force/torque is applied on actuator (see Figure 128).

Effort controller considers more about the torque generated on the joint. As an embedded controller in ROS, it can be used on many robots with effort PID control parameters. Position controller, on the other hand, only focuses on the position of the joints. In this case, since the compliance controller will control the applied force, effort controller increased the difficulties of tuning of compliance controller, therefore only position controller is used in this study.

The controller type must be defined in the “hardware_interface” of the robot description file, “PositionJointInterface” in this case. For the gripper description file, the transmission of the classic position controller must be also carried out. In the main launch file of Gazebo, the controllers should be spawned for two times. The first time is the joint controllers for Gazebo, the second time is the ROS controllers. The two times of spawning must be finished, otherwise there will be errors in launching Gazebo.

For controller parameters in Gazebo, only the parameters should be loaded from the yaml files. To really spawn the controllers, controller nodes should be started. In the definition of the node, the arguments-“args” should be the names of the pre-defined controllers. They should be

“spawned” to the “screen”. The respawn should be false. The “pkg” has been defined as “controller_manager” in the “package.xml” which defines the dependency of the ROS package. It is a C++ class that provides an interface for loading, starting, stopping, and managing controllers. It should be notified that the Gazebo ROS controllers and the ROS controllers should not be the same, otherwise there will be errors spawning the controllers. The motion controller can not only plan for the position of the end-effector, but also the orientation. It calculates the value between actual and target positions using PD controller. The “error_scale” to tune the proportion of the difference in the PD control. For example, increase the error_scale will increase the moving speed of robot, increase the P parameter in PD controller will speed up the movement of robot. “JointTrajectoryController” refers to a type of controller used for controlling the joint trajectory of a robotic system. This controller is part of the ros_control framework, which is a set of ROS packages designed to provide a standardized interface for robot controllers. The JointTrajectoryController specifically deals with the control of joint movements along a predefined trajectory. It is commonly used in scenarios where you want to plan and execute a trajectory for a robot's joints. This type of controller is often employed in applications such as robot arms, where precise control of joint movements is essential.

Compliance controller: in this study, a compliance controller is used to adjust the orientation of the end-effector. The controller is a customised controller to optimise the orientation of the robotic arm according to the recorded contact force and torque (shown in Figure 44). It combines the advantages of active impedance controller, admittance controller and force controller. Thanks to the inverse dynamics feature of Gazebo and ROS environment, the prediction of the force reaction is possible in this platform. The idea is to implement the force prediction directly into trajectory planning of the manipulator. In this study, compliance controller is used to adjust the orientation of the end-effector. The target is set by setting the position and orientation of target_pose in waypoints. Force and torque values are calculated (in simulation) and measured (in reality) with the ft-sensor, under the rostopic of ‘wrench’. When these values are captured, the difference between actual value and the target value, i.e., actual measured force in z direction is -45N, and the target is -50N, the difference 5N with the error_scale will be used to calculate the next target joint position of each joint with the help of dynamic reconfigure, in this case, with Moveit planner. The error_scale is a convenient option to post-multiply the error on all dimensions uniformly. In this study, PositionJointInterfaces is used as compile-time interfaces to control the robot. PD controller is used to calculate the forward dynamics model ¹⁸¹ with the difference. The relation between the position of end-effector and the target force is like follows:

$$F = m(\ddot{x} - \ddot{x}_0) + d(\dot{x} - \dot{x}_0) + c(x - x_0) \quad (55)$$

With this equation, the time-dependent motion x is related with the applied force. In compliance controller, the controller is a hybrid of position controller and force controller. F is the external contact force, d is the damping factor, c is the stiffness. In current study, the controller is position controller, so the damping is not considered. Stiffness is the factor should be tuned in config. The end-effector should be assigned as “wrist_3_link” and the forward dynamics model calculates the joint positions from base link to the end-effector link. Ft-sensor link and compliance_ref_link are assigned to acquire the actual and target wrench force/torque values. The “stiffness” configs how the compliance controller reacts with the difference between actual and target, it is in the unit “N/m” indicating the end-effector moving distance with the contact force. The stiffness is in 6 DOF, functioning with 3 axis force and 3 axis torque. When the stiffness is bigger, it is harder to move the joints. The trans_gain and rot_gain are still there for PD control of the movement of the end-effector. To move the robot with compliance controller, another controller is needed to move the robot to target position, “joint_to_cartesian” controller is used in this case. The design is that joint_to_cartesian has a “son” namespace, which contains the robot joints, in this case, the compliance controller controls the joints in the “mother” namespace while the joint_to_cartesian controller moves the joints in “son” namespace without conflicts. If these two controllers are in one namespace, the controllers with the same joints will be conflicting.

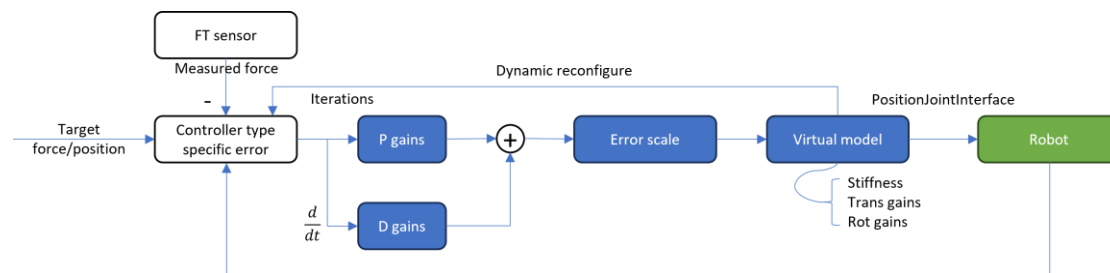


Figure 44 Details of the solvers in the controller.

After setting the controllers, the more important thing is to control the robot with the good planning of using these controllers. Currently, only simple PD controller is used in the compliance controller, it can be optimised in future study to implement PID and adaptive parameters to improve the compliance control.

3.5 Simulation test and the synchronisation with real robot

With the help of firmware in the teach pendant of UR5e, the external control (shown in Figure 45) can be realised (shown in Figure 46). On the dependant of UR5e, design a new program named “External Control”, which contains a step requires external control from the IP of the working computer. Use the launch file in ROS to bring up the real UR5e to the control. During the processing of the code, run the “External Control” test program on the dependant of UR5e.

The feedback of real robot can be detected as shown in Figure 45.

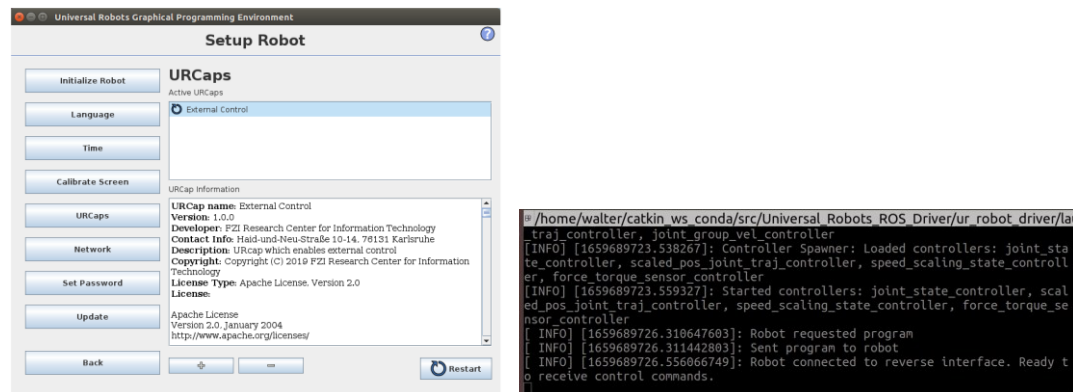


Figure 45 Left: Installation of the 'external control' firmware into the teach pendant. Right: terminal display when synchronisation started.

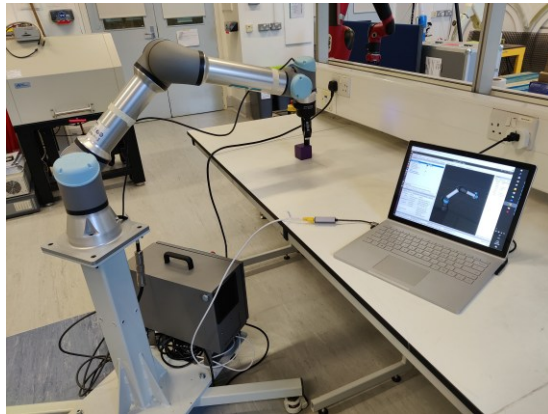


Figure 46 Synchronisation of the simulation model and the real-world robot.

In the point contact scenario, the orientation of the robotic arm can be adjusted by the steps as follows: the robot is prepared with the original position. The robot moves horizontally to the top of the first pre-defined waypoint. The end-effector moves downwards to the first waypoint. After contact, the force and torque during contact is measured, then the robot leaves the surface. The optimised orientation is calculated, the pose of the robot will be adjusted. Then, the pose is optimised, the robot moves back to the surface (shown in Figure 47).

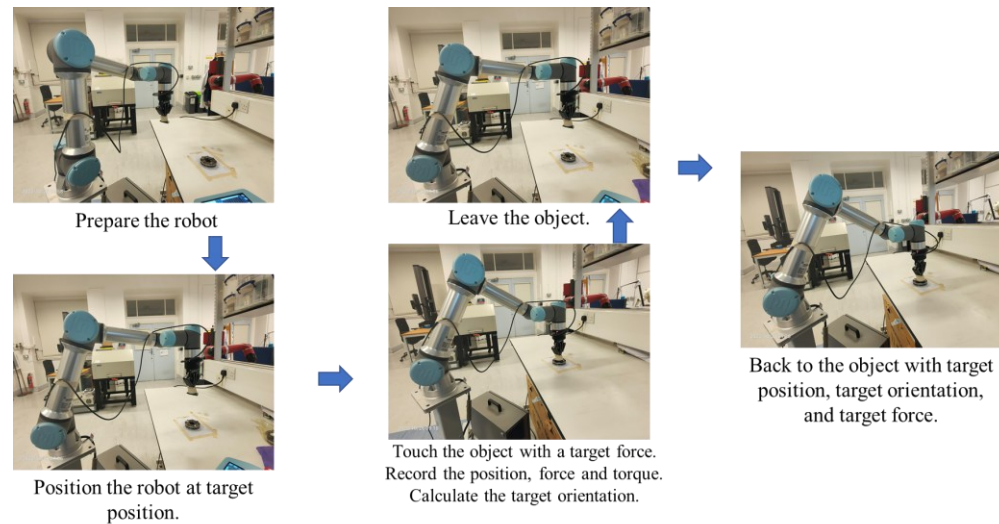


Figure 47 The processes of orientation control of point contact scenario.

Experiment 2(E2): synchronisation of the simulation model and the real world. This is to verify the established robotic platform can control both the simulation model and the real robot. To implement compliance controller, extra settings should be carried out on the controller config. Since the physical condition in the real-world is not completely the same as the simulation environment, the adjustment of the controller should be done. The flat surface scanning in simulation and real world are shown in Figure 48.

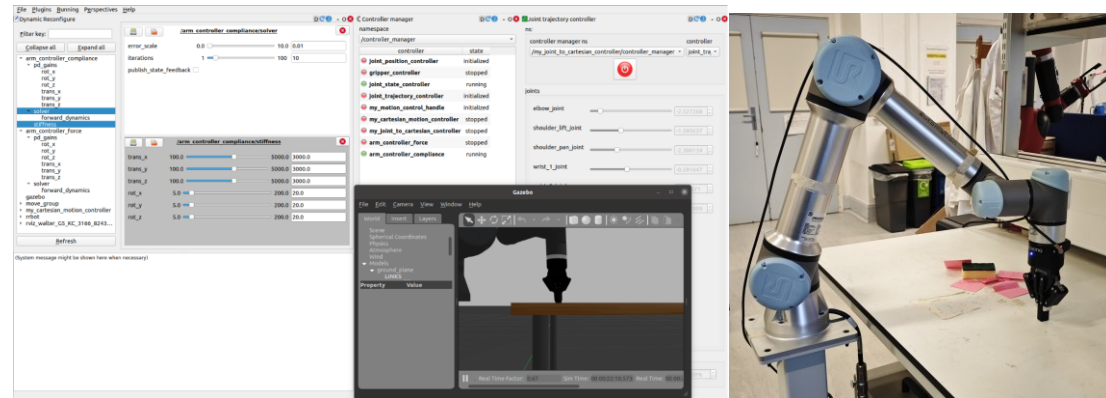


Figure 48 The surface scanning on flat surface in simulation (left) and real world (right).

However, since the real scanning task is implemented on real objects, which are curved or even irregular shapes in some cases. As an intermediate stage between flat surfaces and real target objects, a known, regular curved shape can be set as the object. The compliance controller should implement the force control, orientation adjustment and position movement on the curved surface. The contact force curve and simulation environment can be shown in Figure 49. As can be seen, when compliance controller was applied, the contact force had an oscillation. The issue happened when the end-effector touched the curved surface. The end-effector moves towards the target location via the force controller with target force. When the end-effector

touches the surface and applies impedance force, the compliance end-effector will move passively, which causes a measured force oscillation. After that, the compliance controller will kick in and the actual contact force is approaching the target force and kept steady.

To carry out the curved surface scanning, the detailed processes are shown in Figure 50. On

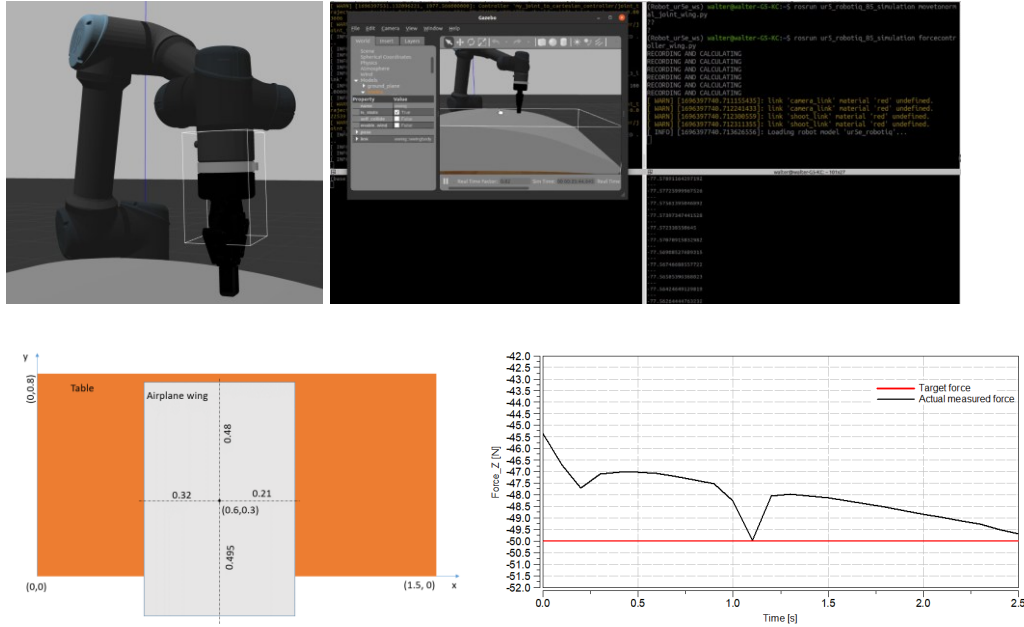


Figure 49 The surface scanning on a curved surface in simulation (upper: simulation environment; lower left: location of the airplane wing on the table; lower right: the curve of contact force during scanning).

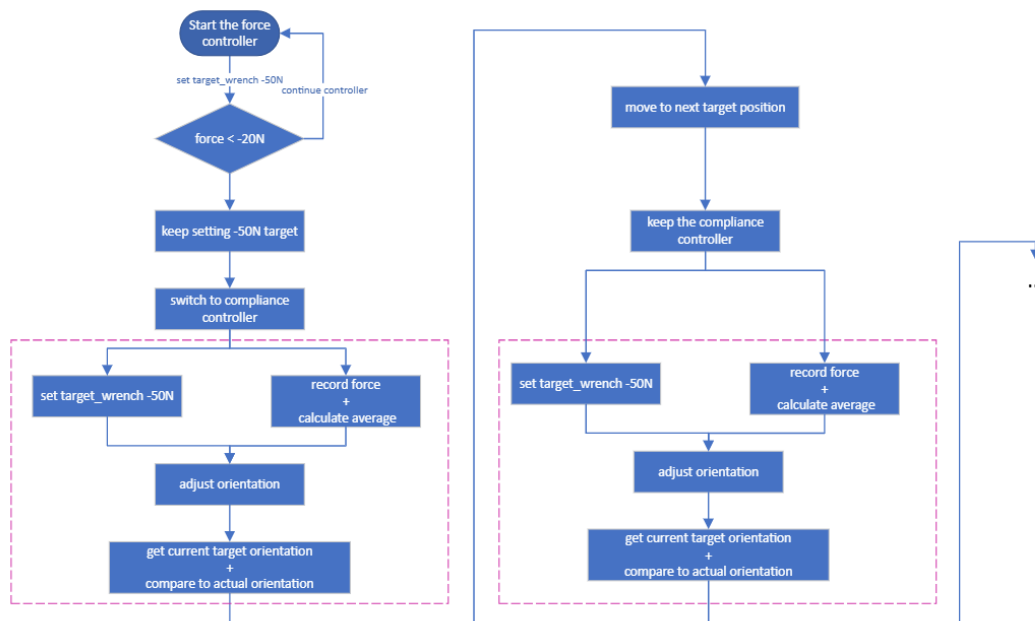


Figure 50 Flow chart of the whole work processes in the surface scanning task.

each targeted position, the force controller should be used to approach the position firstly, after the contact force is larger than 20N, switch to compliance controller, keep the target_wrench at -50N. During the appliance of compliance controller, record the force data and calculate the average and variance for future use. In the next step, use the average force in 3DOF to calculate the actual current orientation. Subsequently, get the current location of the end-effector and use the interpolation algorithm to get the actual normal orientation vector on the target surface. Then, the difference between the actual and target orientation can be calculated. After that, under compliance controller, adjust the orientation of the end-effector and move the end-effector to the next position.

In this case, an airplane wing CAD model is placed on the table. The wing model is in regular curve, which the cross-sectional shape is an airfoil, and the 3D model and the position of the model are known (see Figure 49). The advantage of setting the model is that the actual trajectory can be compared to the real object surface curvature. Only in this case, the controller and methodology of this study can be proved in a transitional scenario. The curved surface is also used in RL training, which will be described in detail in the next section.

4. Control System Design and Implementation

4.1 Control criteria and strategy

Normally, the controllers in ROS are position or velocity based, which can be only used to trajectory to target position, but not apply target contact force. While some other controllers only control the contact force, but never good at trajectory to the target position. While in NDT inspection tasks, adding compliance is essential for this study. For classic external force control on robotic arms, impedance and admittance controllers are commonly used in research. Impedance controller focuses on trajectory following, controlling its stiffness or rigidity in response to external forces or desired trajectories. While admittance controller controls the interaction between a robot and its environment by regulating the robot's compliance or stiffness in response to external forces. Above all, the difference between admittance and impedance control is that the former controls motion after a force is measured, and the latter controls the force trajectory is taken. Since this study needs control both the force and position, the compliance controller combining impedance and admittance should be implemented.

UR5e is a classic articulated robotic arm, which is powered in their joints, and hence require control algorithms to perform continuous mappings from task space to joint space. To transfer from task space to joint space, Based on the kinematic model of the robotic arm mentioned above and the studied questions in this article, it can be observed that there are 3 constraint equations, while the criteria: force control should be within $\pm 5\text{N}$, and the position and pose control should be $\pm 1\text{mm}$ and ± 10 degree.

The controller should be used in normal conditions, which is not limited to the contact scenarios. The controller should be able to control UR5e in simulation and real-world, so the RL training results can be transferred between them. The compliance controller is used as a low-level controller as classic position controller, while RL controller will be used to train the robot to implement scanning inspection actions. Subsequently, the scanning movements will be transferred to the real world, with the trained RL model and the real-world compliance controller (see Figure 51).

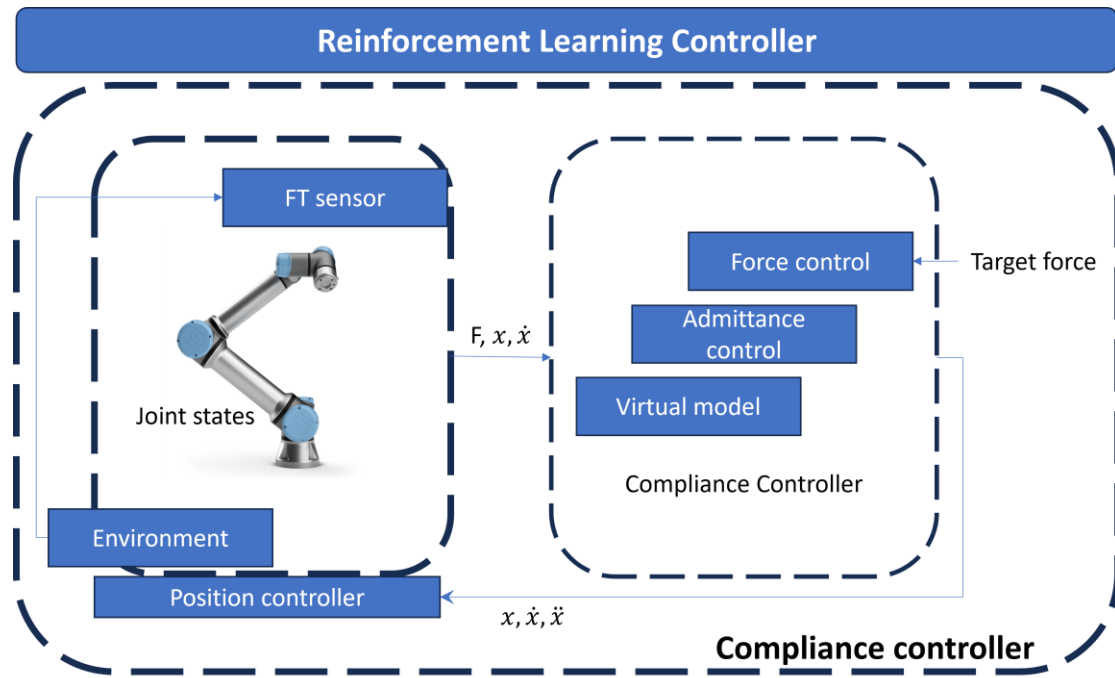


Figure 51 Control strategy of compliance controller.

The control model is based on equations (51) and (55). The external force on the end-effector equals to the compliance force that the controller produces, while the displacement, velocity and acceleration of the end-effector can be calculated based on the reverse kinematic model and dynamics model in Section 3. The goal here is to compute a time dependent motion $x(t)$ of the end-effector in order to control the applied force on the system (Admittance Control). For the general case, a series of joint motion $q(t)$ must be found, which would affect this 6D motion $x(t)$, which is essentially trying to solve this problem from an inverse perspective. With the force/torque (FT) sensor of UR5e, the control loop of the controller can be closed. The controller takes the real measured force and current joint positions and velocities as input and calculate the target joint positions as output to control the robotic arm. The direct solution of the model is the accelerations of the joints, after integration, the velocities and positions can be assigned to joints.

The target of the control is to implement the same controller not only in simulation, but can also be transferred into the real world. Since in the UR5 ROS controller, PID is the most used basic controller in industrial, the advanced PID control parameters are applied to implement the compliance controller.

4.2 Controller design and analysis

To describe robotic tasks in task space, the system should provide user interfaces, which receive commands of task or a target position, while maintaining its compliant behaviour for the end effector. The proposed controller provides interfaces to simultaneously command both desired

motion x_d and desired force F_d with respect to an arbitrary end effector coordinate system (x , y , z). This is possible through the conjunction of virtual Impedance Control and Admittance Control. The characteristics of the compliant behaviour can be adjusted through parameters for the virtual inertia, dampers and springs (see Figure 52).

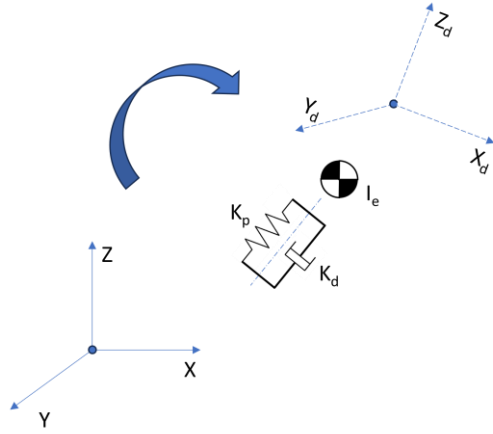


Figure 52 Illustration of virtual spring, damper in force control and target position in position controller.

The virtual spring and damper indicate the PD controller at the inertia of the end-effector. While spring acts like proportional controller, the damper acts as derivative controller. The proposed controller will not only control the target force, but also trajectory the end-effector from original position and pose (x , y , z) to the target position and pose (x_d , y_d , z_d).

$$F_s = c * \Delta x \quad (56)$$

$$F_d = -k * \dot{x} \quad (57)$$

$$F_i = -I * \ddot{x} \quad (58)$$

$$F_{net} = F_{Target} + F_{Measured} + F_i + F_s + F_d \quad (59)$$

$$F_c = F_{ext} = -K_p * F_{net} - K_d * \dot{F}_{net} \quad (60)$$

Where F_{net} is the net force from target force, sensor measured force and all virtual spring and damper force. F_s is the force from virtual spring, which is also proportional controller. c indicates the stiffness of virtual spring. F_d is the force generated from the virtual damper. k is the damping factor. F_i is the force generated through accelerating the apparent mass of the end effector. I is the inertia of the end-effector. F_c is the control force, it equals to the external force F_{ext} . And they equal to a PD controller (see Figure 44). The control force will be calculated at each joint and then the demand position will be calculated using IK solver.

In the controller design, both open-loop and close-loop are considered, since there are scenarios

such as when the compliance controller is applied, but the end-effector is still not contacting with the rigid surface. In such scenarios, the open-loop controller will be used, and it will be switched to close-loop controller when contact is happening. The virtual model in this controller is important, which can be used to calculate the demand position for compliance control. Because the movement during compliance control is not straightforward, especially when applied dynamic forces, the virtual model of the controller can calculate the virtual

Stiffness in each Cartesian dimension. It balances force-torque measurements with motion offsets. The higher the values, the higher the restoring forces (and torques) when trying to move the robot's end-effector away from the commanded target poses.

The controller will set the “wrist_3_link” as the end-effector link. The end-effector link will be set and all the calculations will be implemented based on the TF tree relationship from the base link to the end-effector link.

The compliance reference link will be set again as “wrist_3_link” as the compliance contact force will be measured on this link and also the contact force and calculated acceleration will be assigned on this link.

To implement controller in ROS, it has to be set the joints and PD controllers for the end-effector. The p and d gains determine the responsiveness in each individual Cartesian axis. The higher these values, the faster does the robot move in response to the force-torque and motion inputs. For all UR robots, shoulder_pan_joint, shoulder_lift_joint, elbow_joint, wrist_1_joint, wrist_2_joint, wrist_3_joint are all the links used in simulation and real-world control. In end-effector link, there is a PD controller to make the external force to be 0. PD controllers dropped integral controllers since the control are continuous and integral factor is not used. It increases the efficiency and stability, since it decreases the calculation and without noise disturbance in the system, which is mainly caused by integral term. Integral windup can result in poor performance or even instability of the control system. It can be also prevented by using PD controller.

There is a solver used in the PD control, the solver setups include error_scale, iterations. Error_scale sets the scaling factor for the error in the solver. It determines how much the error affects the control output. In simulation, for example, the scale is set to be 0.01. Iterations parameter specifies the number of iterations performed by the solver during each control cycle. More iterations generally result in a more accurate solution but may increase computation time. In your configuration, the solver performs 10 iterations.

Moreover, it is even more important to have the data interface between the controller and the measured values. In Gazebo, there is a plug-in UR5e simulated FT sensor- wrench. The sensor

can calculate the physics contact force based on all the physics settings, such as the rigidity of the objects. Target wrench is a rostopic that can be used to set the target contact force. The type of the rostopic is geometry_msgs/WrenchStamped (see Figure 53).

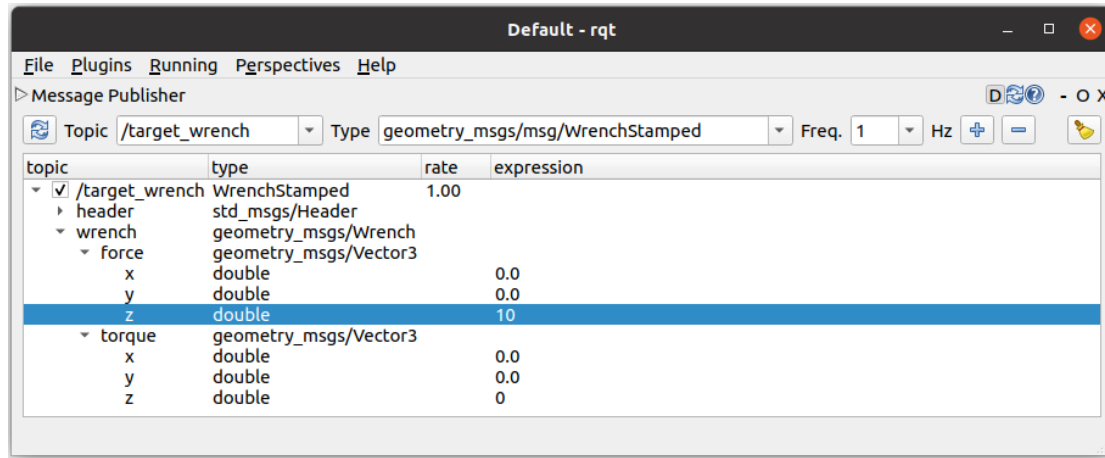


Figure 53 Target wrench rostopic publisher.

To achieve the measured force, a simulated FT sensor needs to be inserted in the urdf file. This involves adding a sensor link and defining the sensor's properties such as its mass, inertia, geometry, and frame relative to the robot's base link. The plugin should subscribe to Gazebo's physics engine to obtain information about the forces and torques acting on the sensor link due to contact with objects or other physical interactions in the simulation. Within the Gazebo plugin, publish simulated force and torque measurements as ROS topics using Gazebo's ROS interface. These topics will contain the simulated force and torque data generated by the plugin, which can be used by ROS controllers or other components in your simulation setup.

The controller is implemented via C++ code in ROS. The pseudo code for the solver part is:

1: **procedure** Solver ($x^d, q_0, \Delta t, N$)

2: $\epsilon_0 = 0$

3: **for** $i = 1$ to N , **do**:

4: $\epsilon_i = x^d - g(q_{i-1})$

5: $\dot{\epsilon}_i = (\epsilon_i - \epsilon_{i-1}) / \Delta t$

6: $F_i = K_p * \epsilon_i + K_d * \dot{\epsilon}_i$

7: $\ddot{q}_i = H^{-1}(q_{i-1}) J^T(q_{i-1}) F_i$

8: $\dot{q}_i = 0.5 \ddot{q}_i \Delta t$

9: $q_i = q_{i-1} + 0.5 \dot{q}_i \Delta t$

```

10: end for

11:  $q_d = q_N$ 

12: return  $q_d$ 

13: end procedure

```

where q_d is the demand motion, q_0 is the original joint position, Δt is the time step, N is the total number of steps. H^{-1} is the inverse matrix of IK transformation. J^T is the transformation of Jacobian matrix. K_p , K_d are the parameters of PD controller.

In this study, the compliance controller will be used for the contact force control, in addition to that, a RL optimised controller will be used to control the trajectory. The target of this study is to combine compliance controller with RL algorithm, since compliance controller focuses on force control and path control, while RL will affect the trajectory planning based on the real-time situation. RL controller will only adjust the moving velocity and orientation adjustment during scanning, therefore, the compliance controller is used as a basic lowest-level controller (see Figure 51). For the RL task, during each training session, the RL controller will be using the force controller approach the trajectory and then use the compliance controller to implement scanning actions. The RL controller will then optimise the parameters during the compliance scanning (see Figure 54).

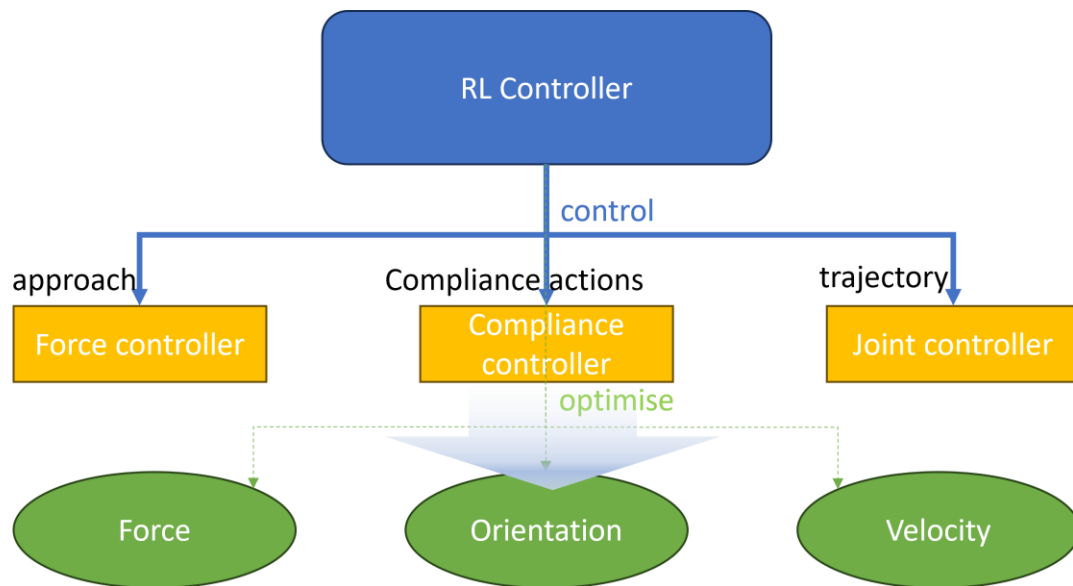


Figure 54 Working diagram of RL controller and compliance controller.

4.3 Simulation and verification

To implement the controller in this study, ROS Gazebo simulation and real-world validation

tests were carried out. To validate the compliance controller, the robot model is implemented in the simulation environment which is similar to the real-world lab. Set a starting point with the x,y coordinates (0.3, 0.4) and use the force controller to approach the table surface. Set target force as force_z is -50N, after the end-effector hit the surface, switch to compliance controller, then keep target force as -50N and start the flat surface scanning.

As shown in Figure 55, the working scenario is firstly launched via roslaunch. It will deploy the world (which includes light, physics properties), the table, the object and the robotic arm. RViz will be also started from the roslaunch, which can be used to plan trajectories, however, python code is used for a series of activities, so RViz is currently not used. After Gazebo is started, a python code will be ran via rosrn. It will guide the end-effector to the position $x=0.3$, $y=0.4$, and make the pose as perpendicular to the table surface. Subsequently, switch the controller to force controller, which will only control the target force. The currently used controllers can be checked in the ROS tools “rqt”, in which there are many more tools, such as joint trajectory controller, TF tree, rostopic curve plot, etc (see Figure 56). In the python code, the condition will be evaluated, if the contact force in the z direction exceeds -10N, it means the end-effector is contacting the physical table surface, then the compliance controller and joint controller will replace force controller. It should be noted that when applying force controller, it is not possible to use joint controller, since the control principle is different. After the compliance and joint controller are switched, target contact force can be set again (see Figure 57) and the end-effector can make the target trajectory to target position and pose.

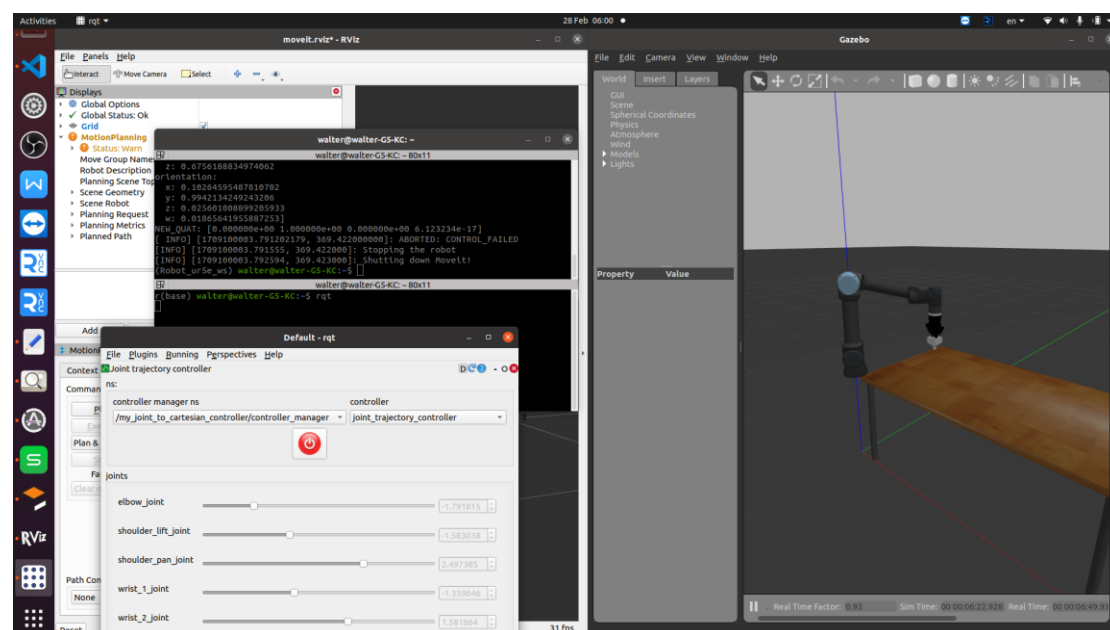


Figure 55 Screenshot of working scenario in Gazebo.

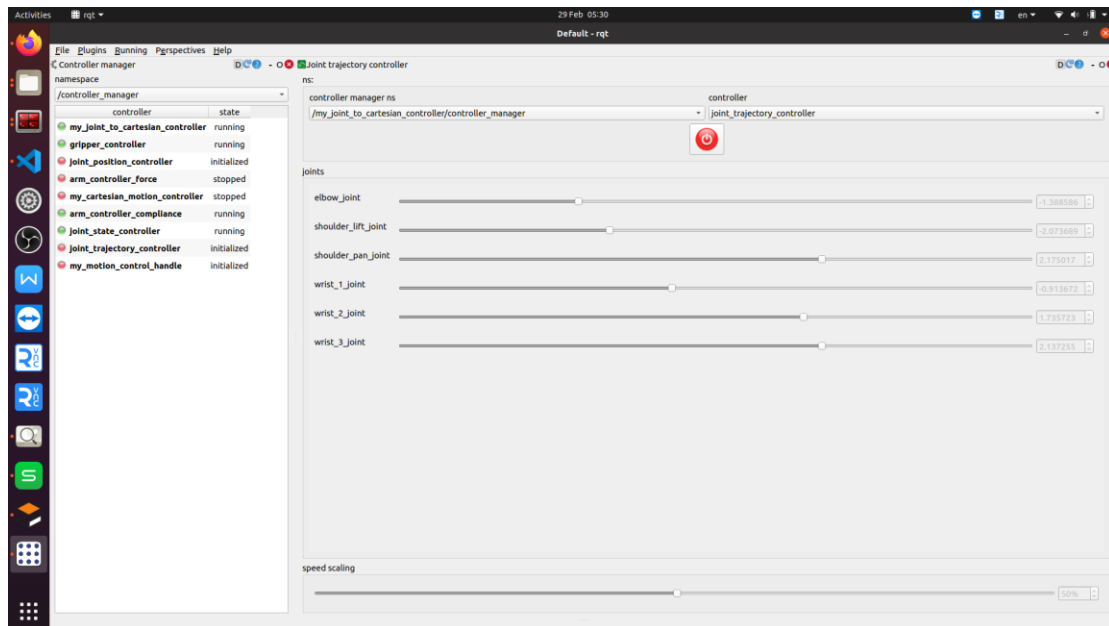


Figure 56 Screenshot of rqt tools in ROS.

As shown in Figure 57, the end-effector of the robotic arm has contacted the surface and it is still normal to the surface. Subsequently, the end-effector can be manoeuvred to target position (see Figure 58). The end-effector moves from position 1 to position 3, the straight line action distance is 20 cm. With the pose of the end-effector staying the same, the end-effector keeps the contact force during the movement.

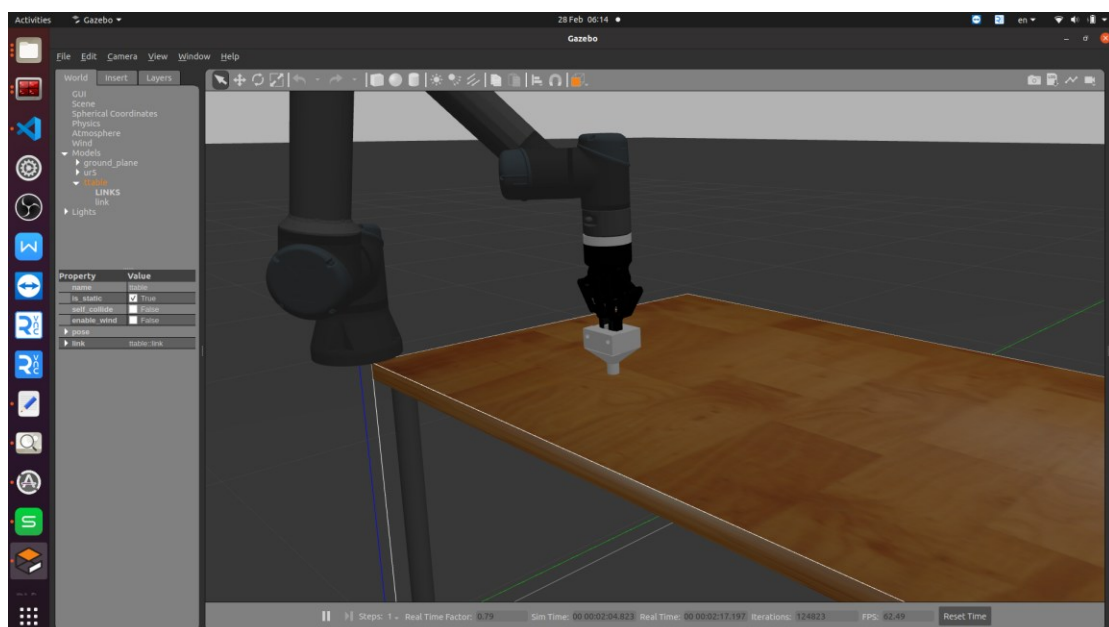


Figure 57 The end-effector contacts the surface of table and compliance controller started to function.

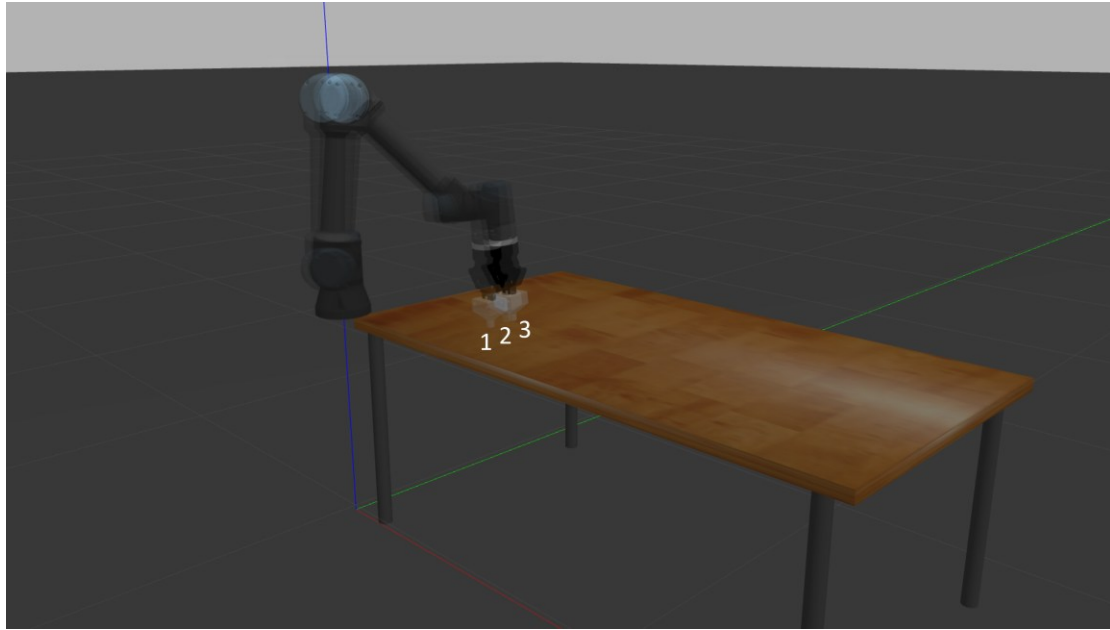


Figure 58 The surface scanning on the flat table surface.

There are some parameters in the controller that need to be modified to tune the controller. As shown in Table 7, the orthogonal experimental method was used in this test. A rough tuning was implemented firstly and then a finer tuning of the parameters was carried out. In the rough tuning, the stiffness of each joint was set the same. Only the gains of “trans” and “rot” of these joints were adjusted. For PD controllers of each joint, they still stayed the same for every joint. Error scale and iteration number of the solver are also tuned to get better results. The average and variance of the control results were logged to compare the results. 30 seconds of the logged data in the stable stage will be used to calculate the average and variance from each log. The final experimental result was shown in Figure 59.

Experiment 3(E3): tuning parameters for the compliance controller to maintain constant contact force.

The data in Table 7 was selected from whole simulation experiments. It can be seen that when the stiffness is too small, the end-effector cannot reach target wrench, moreover, the stability was very poor. Too big stiffness will lead to contact force overshoot. A bigger PD controller parameter will lead to unstable performance. And the selection of iteration in the solver is very impactful on the result. After all the experiments in the simulation environment, the final parameter selection was confirmed for the controller. The average contact force is close to the target wrench, and the variance is small comparing to the results from other parameters.

During the movement, the FT sensor will measure 6 DOF force and torque. As shown in Figure 59, 3 directions of force data were logged. After logging, the statistic is calculated. The average value of contact force is -51.062N , the variance is 2.507 .

Table 7 The comparison of the control parameters of compliance controller.

	Stiffness		PD		Error scale	Iteration	Result	
	Trans	Rot	Trans	Rot			Average	Derivative
1	500	20	0.005	0.1	0.001	2	36.539	26.019
2	500	20	0.05	0.1	0.001	2	51.558	29.310
3	500	20	0.005	0.1	0.01	2	54.557	3.607
4	500	50	0.005	0.1	0.001	2	57.462	6.063
5	2000	20	0.005	0.1	0.01	2	64.691	7.932
6	1000	20	0.005	0.1	0.001	2	57.023	7.118
7	1000	20	0.005	0.1	0.001	10	60.565	7.464
8	1000	20	0.005	0.1	0.001	20	53.839	7.464
9	1000	20	0.005	0.1	0.001	25	51.062	2.507

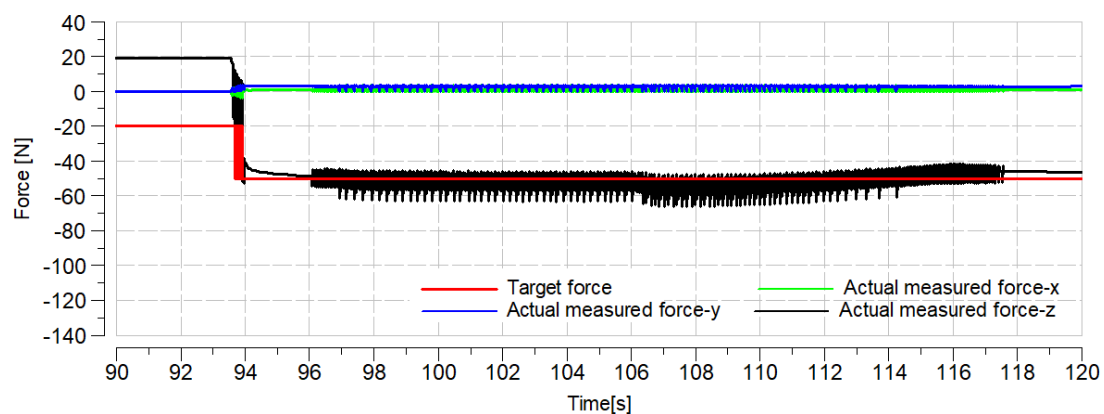


Figure 59 Force curve of the end-effector during movement.

As shown in Figure 59, the contact force will be switched to -20N when it is approaching the surface, using force controller. When contact force is smaller than -5N , the target wrench will switch to -50N . Then, after switch, the end-effector will be moving to the target position (as shown in the vibration section in the curve). To compare the control parameters in the controller, the results can be seen in Table 7. Because the tasks in this study are not related to complex

trajectories, the action space is limited to the table surface. The singularities were not studied in this article.

The contact force was also measured during the transition stage (see Figure 60). Once the actual contact force is smaller than -10N, the target wrench switches to -50N, and the compliance controller tries to achieve the target contact force. The system vibrated for 0.4 seconds, the error was ± 20 N, and then the end-effector settled down.

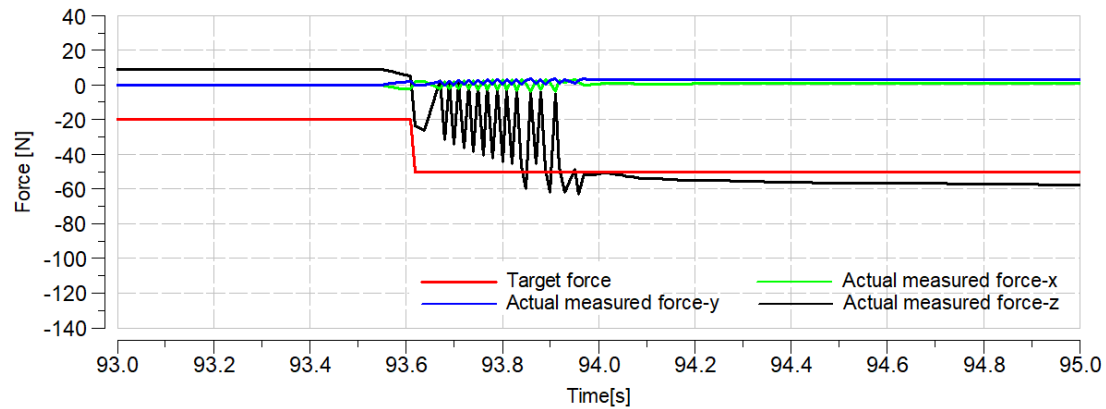


Figure 60 Force curve during the transition phase.

It can be seen, when the actual measured force-z reached -10N, the target wrench will be set to -50N. The reason why the target wrench for force controller is -20N is that the force controller has a bigger acceleration for the end-effector. If target wrench is set to -50N, then the end-effector will have a bigger speed touching the table, which is not safe for the transducer during scanning. The oscillation period took only 0.4 seconds.

To validate the controller, the compliance was also applied on the real arm (see Figure 61). The similar movement was tested, trajectory along y-axis for 20cm. the force data logging curve is shown as Figure 62. The average of the measured force was 48.5 N and standard deviation 1.742.



Figure 61 Photo of real arm using compliance controller.

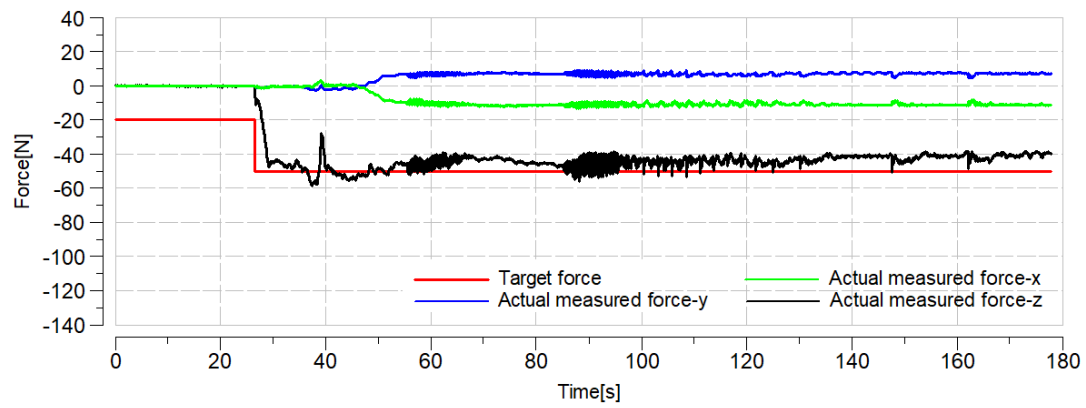


Figure 62 Force curve for compliance controller of real arm.

As can be seen, the real arm performance is not ideally the same as the simulation. But the target force is met, and it can be seen that it took 1 second for the real arm to reach the target wrench, which is close to the simulation.

5. Path Planning, Optimisation of Trajectory Planning

Since the target of this study is to implement robotic surface scanning on unknown surface of the objects to be remanufactured, it is important to implement reinforcement learning (RL) in the robotic scanning process. The reasons to implement RL are as follows: the objects in robotic UT in remanufacturing are often unknown shape problem in industrial UT. And since UT is very sensitive to the movement of the transducer, robotic UT needs the probe to be contacting the surface of the object and a constant contact force is needed to make sure the probe is sticking to the surface to get better reflection of UT. The control of robot which manipulates the probe is very crucial to the inspection results. Therefore, it is a dynamic optimisation problem for the robotic control without any prior knowledge. Comparing to normal machine learning methods, which need massive tagged data that is normally not available for certain tasks, RL can learn from the interaction between the agent and the environment and evolve during training, therefore, it is the choice for this problem. Since UT is a mature technique in industry, the UT quality is not considered in this study.

Robotic arm is good at implementing repetitive tasks, UT scanning in manufacturing industry is a suitable job for it. Since the UT in industry is straightforward, no complicated path planning is needed. And since the objects in industrial UT are often rigid body, while the target in medical UT is soft object, such as human body, the dynamic contact force and deformation problem is easier to solve. The only need for robotic tasks is to plan the control of the robot according to the profile of the object. However, the UT on industrial object is still challenging, it is a multi-objective control, i.e., the contact force, the orientation of the end-effector and the overall scanning trajectory should be all evaluated and controlled in the real-time.

Since the components under remanufacturing are within various types of shapes, furthermore, the components under remanufacturing are mostly used for a long time or under heavy loads, the shape and size are different from the original one. So, the planning the trajectory of robot with original CAD model is unavailable, for example, in die casting, the average experimental tool wear is over 1.5mm¹⁸², which is enough to affect UT inspection. Even with CAD model, the control of robotic scanning at extreme situation is challenging, for example, at the edge of the object. Computer vision was planned and tested to reconstruct the object and plan the path, but since most of the industrial components are made of metal, the reflection of illumination affects the results of CV reconstruction results. Other metrology method, such as, 3D scanner, can be implemented to achieve robot trajectory path, however, the equipment needed is expensive, and the operation of equipment is experience-based. The mis-operation will lead to noise in result and result in errors or distortion in robotic trajectory. The reflectivity of the surface and the condition of environment also have effects on the results of metrology.

Therefore, contact-planning, which is similar to manual UT, is planned in this study, as it saves extra sensors and is not influenced by ambient environments ¹⁸³.

Several methods in prior literature have been considered for controlling algorithm consideration (see Figure 63). The traditional PID control is easy to implement, but it is unsuitable for unknown surface scanning, since it can only adjust the contact force, and the overshoot always happens in minor actions. For orientation, it has no capability to control it with multiple inputs, i.e., forces and torques. Moreover, since PID controller only has one set, or more sets of linear model control tuning parameters, it is lack of interaction with the special environment ¹⁸⁴. PID is also difficult to handle real-time tasks, therefore not suitable for non-linear complex tasks.

Fuzzy control can be used combining the PID control or other algorithms, but there is probability to malfunction. And the drawbacks of fuzzy control are the randomness of the control, the sensitivity to disturbance and the limitation in complex system ¹⁸⁵.

Model predictive control (MPC) can be also used to control orientation of robot. However, the tuning of MPC parameters has been always difficult and like a black box. The calculation complexity makes the time delay, and this may cause communication problem. MPC is also sensitive to disturbance, which will increase the instability of system.

Sliding Mode Control (SMC) can be also used for the task, it is used to control the system along the sliding surface and is well-known of its robustness of performance. However, the tuning of SMC is also experience-based, and it often causes chattering in robot trajectory ¹⁸⁶.

Optimisation algorithms, such as, particle swarm optimisation (PSO) is also used in orientation optimisation tasks. PSO can provide decent results, but it suffers from initial conditions, such as, the positions and velocities of the particles. The algorithm depends heavily on hyperparameters, but finding optimal parameters has been never a simple job. Traditional machine learning combined with computer vision has been tested, but since the machine learning needs a large database to train itself, it is not suitable for what we are expecting, a quick adaptive, quick capable solution for a certain task which is like a black-box of scenario. The overall brief research topics bibliographic coupling network on robotic orientation optimisation in Web of Science and Scopus database was built using VOSViewer (shown in Figure 63).

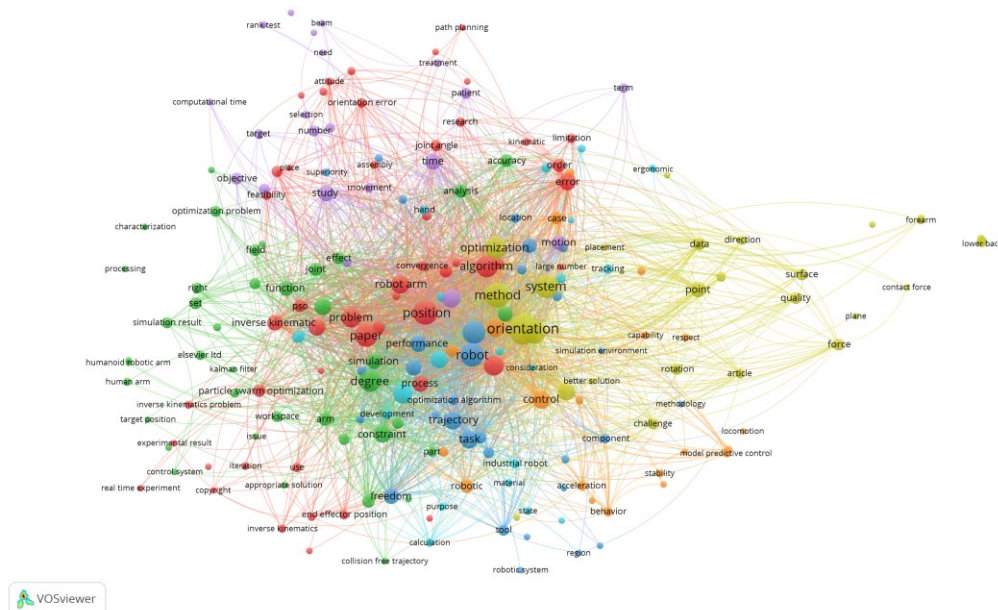


Figure 63 Network of literature on orientation optimisation of robotic arm.

5.1 Path planning for robotic scanning

The main topic in this study is the optimisation of trajectory planning, but path planning is the basement of trajectory. The path planning is not studied, so it is set to be a simple strategy. However, it is still not easy to plan the path of robot for any task, for example, the UK scientist is still working on landing on the moon, which is far more difficult than this study, but still, the path planning on an unknown object is always difficult.

For the path planning on the topic of UT or additive manufacturing, there are some classic path pattern used: raster, spiral, zigzag, hybrid, grid, triangles, etc ^{125, 187}. There can be possibilities to study energy-saving path planning strategies in future study, however, due to the limits of the article, classic path planning methods are used in this study. Raster path pattern is based on planar ray along one direction, it is simple, efficient, and robust to cope with any boundaries. However, due to the difference in the shapes, classic raster cannot be used in the scanning of dies in this study, the circle (spiral) path is used. The selection of path can be also studied in future studies.

To make the path planning simple for the scanning task, it was planned like this: The target object will be located in the assigned workspace (object detection can be implemented but not for now). Based on the specific situation, the starting point location of the scanning can be assigned based on the location of the object. The starting x,y coordinates of the end-effector will be assigned to the robot. Then, the z coordinate will be confirmed by using compliance controller. The controller will bring the end-effector to the surface of the object and apply target

contact force, when the target force is met, the z coordinate will be the final starting position. After this starting position, a series of path points will be assigned to the robot according to the shape of the object (shown in Figure 64). In this study, raster is assigned for rectangle shape object, and circle path is assigned for circle shape object.

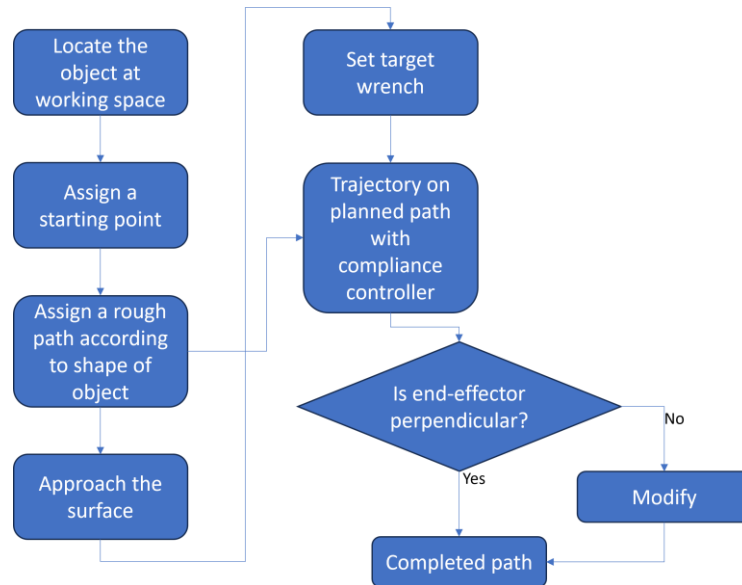


Figure 64 The raster path planning on wing surface in simulation

For example, in the task of scanning on a curved wing surface (shown in Figure 65), raster path planning is used. To scan the biggest area on the wing, the starting position is set to be $(x=0.3, y=0.4)$ of the base link of the robot. The end-effector of robot is set to be perpendicular to the x,y surface to approach the object.

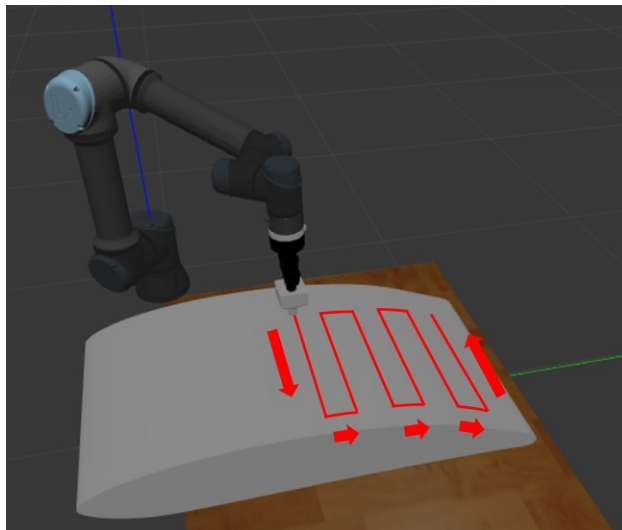


Figure 65 The raster path planning on wing surface in simulation.

As described, the path is not completely planned, as the compliance controller will force the end-effector to stick to the surface of the object. Therefore, the z coordinate is not planned pre-

hand. However, the x, y coordinates can be planned since it is the path points. During the implementation of these path points, the compliance controller will plan the z coordinate dynamic. For the die object, the scanning path of radius of 5cm can be planned (see Figure 66). For future research, it can be implemented that shape recognition and adaptive path planning.

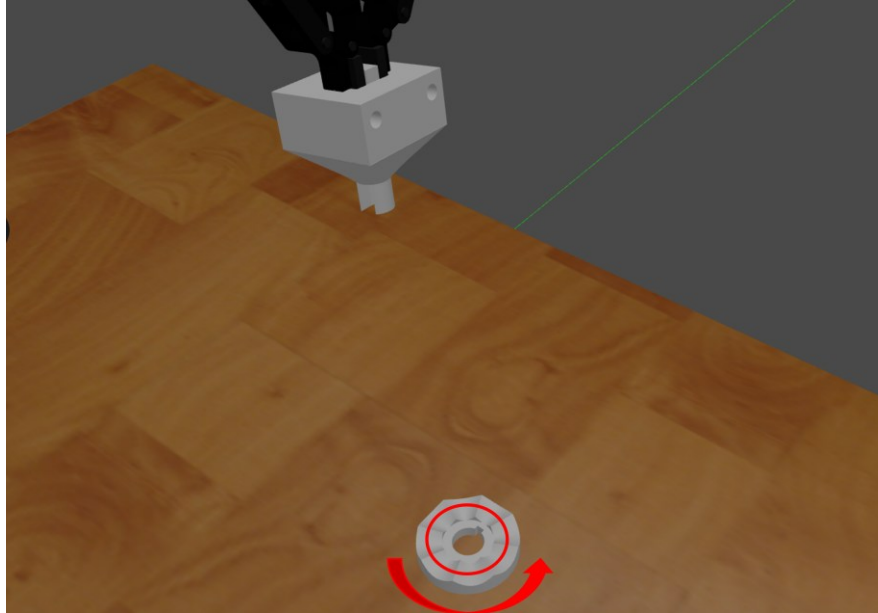


Figure 66 The circle path planning on die surface.

For future reconstruction work, it can be done as follows: since the position of the base_link is fixed. The position of the end-effector will be calculated by the base_link position and the end of wrist_3_link. As the end-effector moves, the position of the tip of the end-effector will be logged. Since the feature of the compliance controller, it applies contact force to the surface, it makes the end-effector stick to the surface, the trajectory of the tip of end-effector can be reconstructed as the surface of the object. The target contour and actual trajectory are shown in Figure 67.

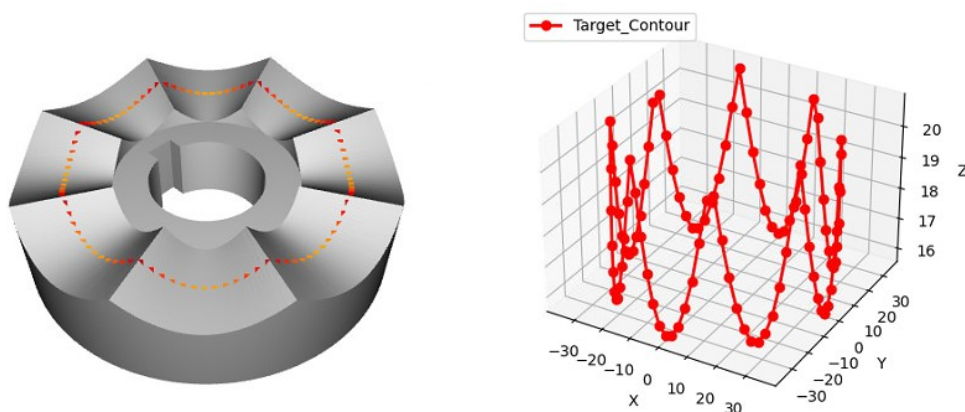


Figure 67 The surface contour of the die surface, left: trajectory points on die surface, right: coordinates

of contour.

As can be seen from Figure 67, after the simulation, the actual trajectory is deviate from the target contour, i.e., the actual points on the die surface. The average difference between the actual end-effector trajectory and the target contour is 0.5506mm, the variance is 0.4287. The difference is acceptable, and it can be used in future research to make a 3D reconstruction of the object combined with UT results.

5.2 Implementation of reinforcement learning with ROS

At first place, it should be considered where to implement the RL algorithm. It is also about why RL should be used. In some prior literature, the application of RL was not clearly explained¹⁸⁸. Therefore, the details of RL should be interpreted in this work. Someone will suspect the use of RL, for example, if we check Figure 50, the robot can work well on curved surfaces without RL. However, RL is not for simple jobs, when the task is not only simply scanning but to adjust on different shapes and even unknown shapes, RL can be implemented. For example, in the intermediate stage, a target is set for the searching scenario, a target position is set for the robot that if the end-effector reaches the target position or gets closer to it, the reward will increase. This strategy can be also used in real UT scanning. In the real task of surface scanning, the scanning velocity selection and the dynamic orientation optimisation should rely on RL algorithm to adapt on different objects (shown in Figure 68), since it is almost impossible to manually tune the control parameters.

In this subsection, a first-stage RL task with robotic arm will be realised to reach a target in the workspace. To implement RL, the whole RL structure should be understood. At first, the movement to be trained should be a Markov decision process (MDP), in which the previous action selection will have influences on next-step actions. In our case, the surface scanning using UT is a very good example of MDP, since the whole process is a continuous, dynamic action sequence. The actions, in this study, is the movement of the robotic arm. The robotic arm will interact with the environment around it and the reward will be evaluated according to the status of the interactions. The environment provides a scalar feedback signal, i.e., reward, to the agent after each action. The reward indicates how well the agent is performing in achieving its goals. The “agent” in this case is the robotic arm, the environment is the dynamics and physics of the robotic arm system. It defines how the state evolves in response to actions taken by the RL agent. The environment also provides feedback to the agent in the form of rewards, which the agent uses to learn. The states are the current situation or configuration of the robotic arm. The RL agent receives observations from the environment, which make up the state. Observations are used to determine the system's current state and are crucial for decision-making. These could include information such as joint angles, end-effector position, velocity,

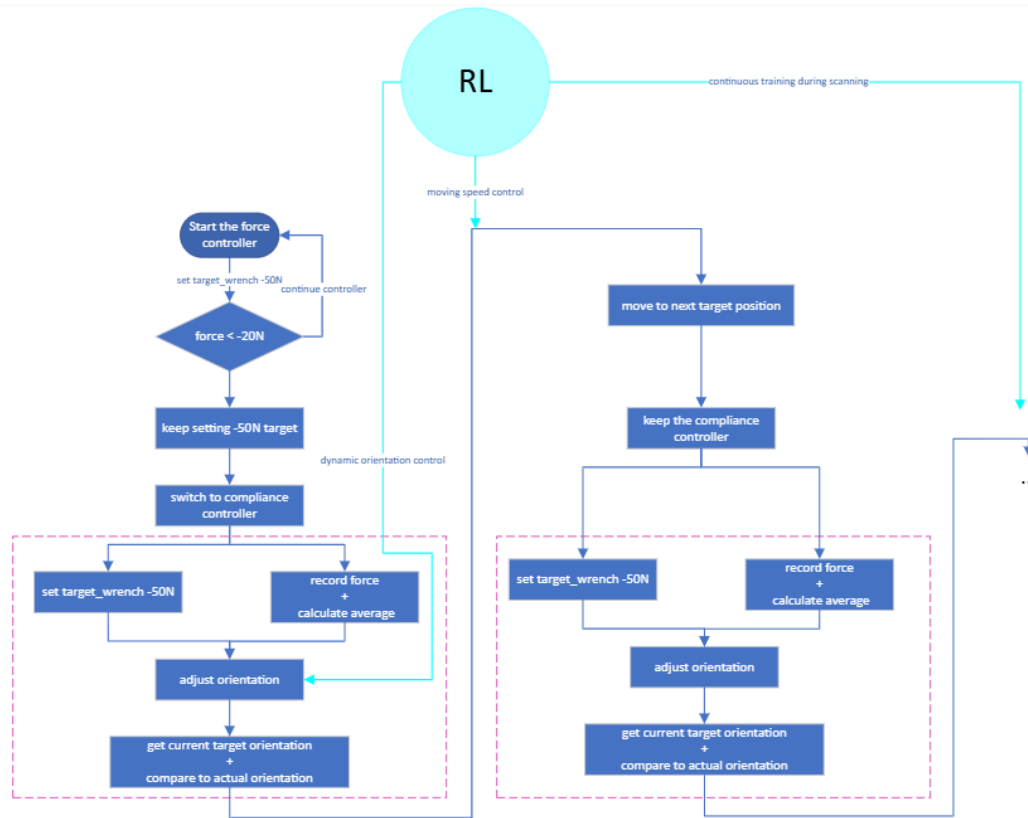


Figure 68 The implementation of RL in the flow chart.

and other relevant parameters of the robotic arm (shown in Figure 69). During scanning, all contact forces, transducer orientations, and moving velocity should be considered to maintain a good UT inspection. These parameters will be the criteria for calculating rewards in RL.

In this study, the combination of ROS+ Gazebo + Moveit + OpenAI_ros + Stable_baselines3 packages is used with Python code. This will be used to establish the agent and environment. The ROS is the middleware of the simulation model and the controllers. Gazebo is the simulation environment, within which the robotic arm can move with physical features and interact with environment. Moveit is a solution for the inverse kinematic model, in which the links and joints of the simulation model can be defined, and the method used for inverse kinematic can be defined. OpenAI_ros is a package created by the ‘Construct’ company from Spain^{189, 190}. The package can communicate between ROS Gazebo and Gym environment from OpenAI company. The advantages of the package are that it has interface with pre-built ROS environment, instead of Mujoco. Since in surface scanning case, customised controller should be used in simulation and real-world to make the robotic compliance with the surface, Mujoco cannot do this. Mujoco is a light-weight simulation software which is more focused on RL training in simulation, however, it is difficult to build customised controller and implement

control in the real-world. OpenAI_ros has a clear RL structure, i.e., robot environment and task environment. It has multiple modular and examples to reference to build customised environment. It is compatible with high-fidelity simulation. Stable_baselines3 packages are a commonly used reinforcement learning algorithm library. It has up-to-date RL algorithms library and interfaces to gym environment and tensorboard evaluation tool. There are many codes of the latest RL algorithms available in this library. The overall proposal of RL in this study is shown in Figure 69.

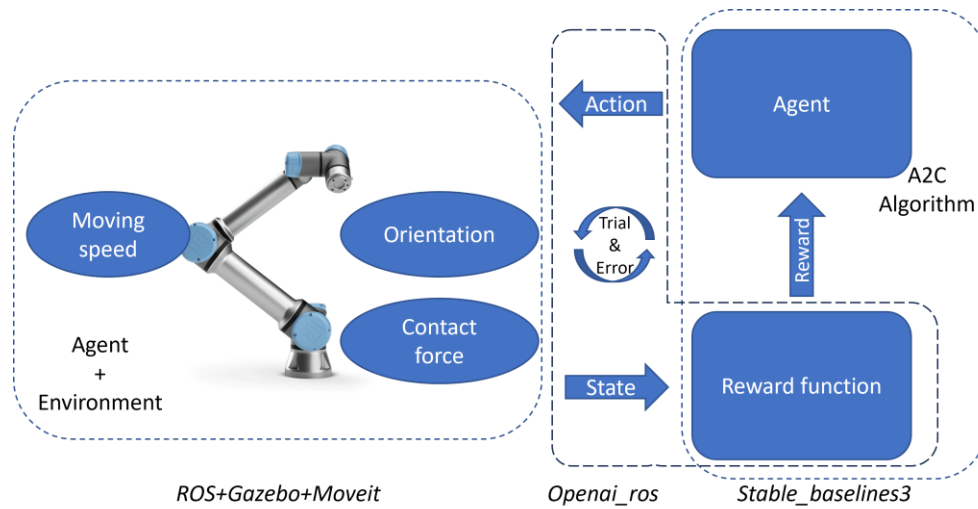


Figure 69 Structure of implementing the trajectory optimisation algorithm using RL.

The structure of realising RL with openai_ros is shown in Figure 70. The main code of RL in openai_ros uses OpenAI gym to build training environment and stable_baselines3 library to choose RL algorithms to train the agent. Gym library is created by OpenAI company. It has been updated to Gymnasium in 2021. It is an open-source Python library for developing and comparing reinforcement learning algorithms by providing a standard API to communicate between learning algorithms and environments.

The simulation environments include classic RL scenarios, such as, robot problem and video game problem. Each environment has a well-defined API with methods like reset (to start a new episode) and step (to take an action in the environment). It serves as a platform which defines the agent, the environment, the action spaces and observation spaces. With these settings, different RL algorithms can be implemented and compared upon this platform. The detailed settings of trainings and configurations of algorithms are designed in the training script. In this study, ROS is linked to gym via API interface. After the launch of the code, the gym environment for the robotic arm is registered via the registration function in gym.envs.registration library. On the other hand, stable_baselines3 library is a set of high-quality implementations of reinforcement learning algorithms in Python, built on top of the PyTorch deep learning framework. It has the compatibility with gym library and Pytorch.

Except for algorithms, `stable_baselines3` also has the evaluation, callback system, logging monitoring, multi-processing, etc.

The whole ROS package is still launched by a `roslaunch` file. Within the `roslaunch` file, an overall python code will be included, which will also link to the task environment. The gym environment is launched in the launch file and the main training python file at first. With the help of gym library, the working environment can be established. “gym” is an open-sourced library launched by OpenAI company. In gym, all the reinforcement learning algorithms can be used in the common ground called “environments”. However, the environments do not have accesses to other simulation environment, such as Gazebo. It needs the interface like `Openai_ros`, which will build gym environment with detailed settings in ROS. When training starts, the script loads the task environment in `Openai_ros` firstly.

In task environment, there are two main script files. One defines the rules of the RL, e.g., reward function, loss function. The other file defines the detailed actions and the `reset_world` movement which also includes the interactions with Gazebo software. The robot environment is loaded within the task environment. The settings of the robot are defined in robot environment. Subsequently, the RL code loads the robot environment, which contains the launch file loading the robot and world and the original position and orientation of the robotic arm. In the robot environment, it provides the complete integration between the Gazebo simulation of the robot and the OpenAI algorithm environments. Finally, the robot description `urdf` file is loaded in a launch file. The Gazebo environment will connect the tasks to the Gazebo software. The Gazebo environment is started at the start of a training episode, it will be stopped at the end of all the training episodes.

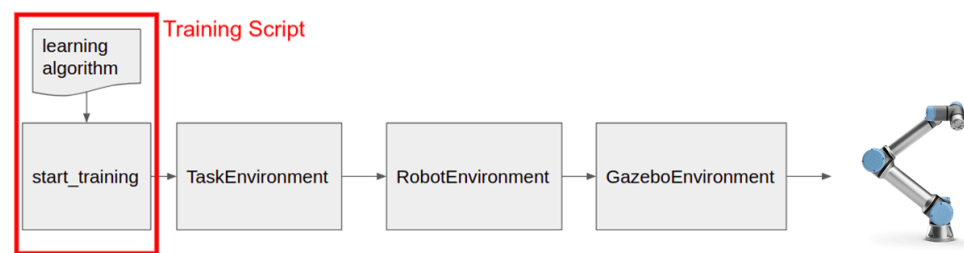


Figure 70 Structure of `openai_ros` package realising RL in ROS.

To implement RL in robotic tasks, a simplified task with a UR5e robot is implemented at first, which is only for the robot to reach a target goal using RL. In this study, task environment is always defined as “move”, within this environment, a class defining the task environment is coded. At first, the directory of the robot environment is defined. With this launch file, the objects in the world which the robot will simulate in is started, currently, the robot model is not spawned. The parameters of the RL parameters in `yaml` file are also loaded via loading the `yaml`

file. In the yaml file, the moving space limitations of the (x,y,z) coordinates of the robot are defined. The robot will not move out of the space.

In the setting of RL parameters, the delta movement of each time episode (moving speed) will be defined, for example, 0.05, means in each time step, the certain joint of the robot will move for 0.05 m. This parameter will also determine the resolution of the movement in the task. In this case, discrete action space and observation space are defined using `gym.spaces`. The number of actions is defined here, for example, if we define movement along z-axis and the delta displacements are 0.05 and 0.1, then they are two actions. Each action should be defined specifically, and the total number of the actions must be consistent with the real number. For the target searching task, the “acceptable_distance_to_ball” is another important parameter that can determine, it is used to evaluate if the agent has reached the target. If the real distance between the end-effector and the target ball is smaller than this acceptable distance, it means the robot reaches the target, the “done_reward” can be earned by the agent. This is also related to the difficulty of the tasks. The reward mark when moving closer to the target and reaching the final target will be also set. The observation is acquired to calculate the reward, the observation is the location of the end-effector. With the location information, the distance between the end-effector and the target ball can be calculated.

In this study, a special code file is written to define the detailed actions. The feature of this study is that the RL can be implemented in the Gazebo environment, which has not been used by many researchers. To demonstrate and test the methodology, a simple environment was established at first. A red cricket ball is fixed at (0.2, 0.2) on the ground of the simulation world, the robotic arm is set to have two actions, which are moving the gripper up and down to reach the ball (shown in Figure 71).

The overall main code is a training script which contains the interface with all “environment”-s. With the integration of `gym` and `stable_baselines3` libraries, it can implement the RL algorithms from `stable_baselines3` library into the `gym` environment. The main code initiates the node for the RL training (“ur5e_train_all” node in Figure 71), it uses `gym` to start OpenAI environment. `gym.make` is the main function that is used. For example, `gym.make("CartPole-v1")` creates an instance of the `CartPole` environment, which is a classic control problem in which the goal is to balance a pole on top of a cart by moving the cart left or right. The `gym.make()` function returns an instance of the environment class, which provides methods for interacting with the environment, such as `reset()`, which resets the environment to its initial state, and `step()`, which takes an action and returns a new observation and reward. In this study, before making the OpenAI environment, the number of max episodes steps is defined in the “RegisterOpenAI_ROS_Env” function. When applying RL algorithms, such as PPO, it can be

more efficient to train agents on multiple environments in parallel. In such case, ‘DummyVecEnv’ class provides a simple way to create multiple copies of an environment and manage them together. It is a class from stable_baselines3 library, which vectorized training, where agents can take actions on multiple environments at once and receive feedback in parallel. The number of the parallel environments can be defined with the “num_envs” function.

Experiment 4(E4): tests for implementing RL in ROS simulation. This is to verify the proposed method can be used to communicate between ROS and RL libraries. For the task environment of the ball searching task, the definitions of RL algorithm are set up in a code file. For example, the set of actions is defined here, the actions include moving up, down, left, right, forward and backward. For each action, the delta distance can be defined in yaml config file. The reward of RL is calculated according to the distance between the end-effector and the target. The reset of the world can be defined using the initial states of the joints. For details of movement: it defines the action space, which has the x-axis limitation from -0.5 to 1, y-axis: -0.5 to 1, z-axis, 0 to 1. “move_tip” function which contains how to move the end-effector. In this study, move_tip only move the position of the end-effector tip, but not the orientation. In this searching case, it can change the moving speed of the robot, modify the factor of orientation adjustment. Reward definition: if it moves closer to the target, reward +10. If the agent reaches the target, the reward +100. The reset action rules are also defined in the task environment, when the agent reaches the target, it should go back to the home position. Other functions, such as distance calculation between target location and tcp, if the tcp is in the working space, are defined.

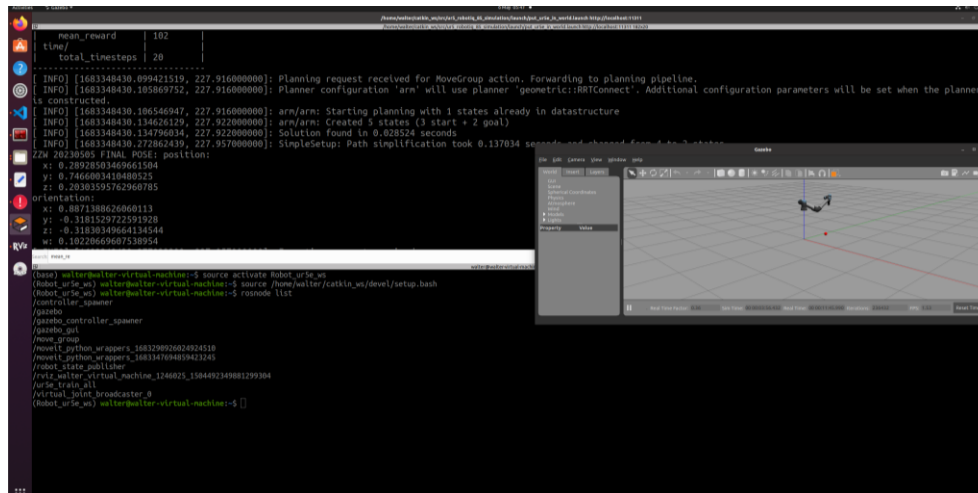


Figure 71 Screenshot of UR5e in Gazebo simulation using RL algorithm.

For the robot environment, the main function of this environment is to link the robot Gazebo simulation and gym library. Moreover, the more detailed setups or special actions for customised robots in the task environment can be also linked to this environment, so everything can be linked in the chain. In the definition of the robot environment, the robot model should

be spawned into the simulation world firstly, which is realised via a python class.

A class ROSLauncher is used to launch the Gazebo environment. This class function is the interface between roslaunch and gym environment. Then, an entry point to the Gazebo simulation should be set in the function. This entry point is the interface between the robot environment and the task environment. The initial settings of the robot environment should be loaded, for example, the “reset_controls” can be set to “False”, then when the initialisation of the robot environment, the controllers of robot will not be reset. The reset control function should be also set with extra care, during reset, if a related controller has been stopped, it should be switched on again after the reset. Otherwise, the related control function cannot take place. The RobotGazeboEnv class function is inherited from gym.env, which defines the standard interface that an environment should implement. It includes methods such as reset, step, and properties like observation_space and action_space. Besides launching the robot description file, the class also defines the initial states of the robot, i.e., the joint states. The class also defines the actions, such as check all the sensors in the robot. The limits of the joints positions should be set carefully, if the limits are too small, the robot will not move to the target location. The position limits of “elbow_joint” in this study are set to $[-360^\circ, 360^\circ]$ in the robot description file. In the robot environment, the function of task environment can be imported, for example, the initial state of the robot.

The initial state of the robot environment is defined to make the robot original pose, which include the start of launch file, the load of config parameters. The working directory of the robot is defined in the robot environment. The states of all sensors and all joints are checked in robot environment. Subsequently, the Gazebo environment will be started, the movement of the robot will be loaded. After the finish of the task, the Gazebo environment will be paused and the loop of task will start again till the training target is reached.

For the OpenAI-ROS Gazebo simulation environment, a gym RL simulation environment should be established in Gazebo using gym.make function. When coding, the max_episode_step can be defined for the gym make process. “max_episode_steps=10000” sets the maximum number of steps which allowed the agent in an episode of the registered environment to reach the target. Episodes are typically used in reinforcement learning, and limiting the number of steps helps control the duration of an episode. During training or evaluation, the environment's step function is called at each time step. The episode continues until one of the following conditions is met: the agent reaches a terminal state (e.g., goal achieved or failure). Or the maximum number of steps specified by max_episode_steps is reached.

The main code of RL training of UR5e can be shown in Figure 129 in Appendix B. In the first 18 lines of the code, related libraries and functions are imported to the main code.

Stable_baselines3, openai_ros package are imported. The working directory is assigned to the directory of the code located. The training model is defined, as in this study, the method used is PPO. With the “StartOpenAI_ROS_Environment” function, the initial gym environment for the robot in Gazebo can be established. The “MlpPolicy” is used for RL tasks except for images, CnnPolicies are for images only. In each RL training, there are time_steps and time_episodes. Each time_step is structured with pre-defined number of time_episodes. For example, if 100 time_steps with 1000 time_episodes, the RL agent will be trained 100*1000 times. During this time, the progressive reward of each time_step can be evaluated. Except for the code of RL training, the limits of training are also very important (see Figure 130 in Appendix B).

In the config file, the most important parameter is the “acceptance distance”, this parameter is to define how the action of the agent can be evaluated as a “done”. The number of decimal precisions should be also set up in config, this will affect the calculated distance between the end-effector and the target location. The training of RL agent can be ended by several methods, for example, the most normal method is to define the total timesteps of the training process, once the total number of training time steps is reached, the training will be ended. Another method is to set a certain level of reward, when the reward is reached, the training will be ended. However, since the reward can be random and sometimes it gets very outstanding, the reward-stopping criterion is not used in this study. The total_timesteps parameter in the learn method specifies the total number of training steps that the algorithm should perform during training. The actual meaning of a "timestep" can vary depending on the environment, but it generally corresponds to one interaction of the agent with the environment. Every time when the agent reacts with the environment, it is a timestep, and an episode can be a series of time step. When the episode ends, the agent will reset to original position and start over again.

The training data is recorded by the code “tensorboard_log”. In stable_baselines3 library, extra setup of logging data should be made. Data call back should be defined as a list. “n_eval_episodes” and “eval_freq” are defined to evaluate the performance of agent. The n_eval_episodes parameter is an integer that specifies the number of episodes to use for evaluation of the agent's performance. For example, if the n_eval_episodes is set to be 10, the evaluation will be carried out in 10 episodes. During evaluation, the agent's policy is used to control the environment without learning, and the resulting rewards are used to assess the agent's performance. The average reward during the 10 episodes will be calculated. The time spent to complete each episode is also recorded.

The eval_freq parameter is an integer that specifies the frequency at which evaluation is performed, in terms of the number of training episodes completed. In this case, evaluation is performed every 100 episodes of training. In the setting of RL, “max_episode_steps” is a

parameter that is often used in reinforcement learning to set a maximum length for each episode. It specifies the maximum number of time steps that can occur during an episode before the episode is terminated. If the conditions of termination of an episode are met, the episode will be finished earlier than plan. So, as the survival of the agent becomes easier, the length of episode should decrease. Another problem happened during the simulation in Gazebo, that the simulation sometimes failed due to error “the start motion planning tree could not be found”. Therefore, a reset call back was defined during the training, to reset the Gazebo environment every time when the time_step ends. To validate the code program of RL, testing over 24 hours has been carried out. To visualise the log data, use the command “tensorboard --logdir ” with the directory of the log data. The visualisation of the log is shown in Figure 72. It shows that the mean reward of training episode is increasing along the time steps. It can be seen in Figure 73, that the end-effector is approaching to the target ball using “moving-down” action. In this study, since the target is an “overall” target, such as, reaching target, the reward curve along episode makes more sense.

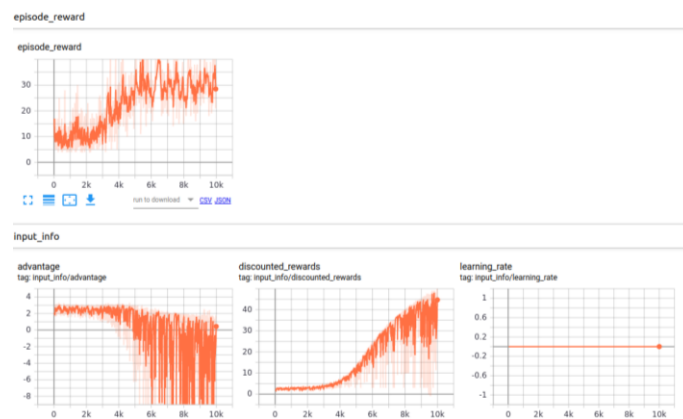


Figure 72 Simulation result for the RL algorithm.

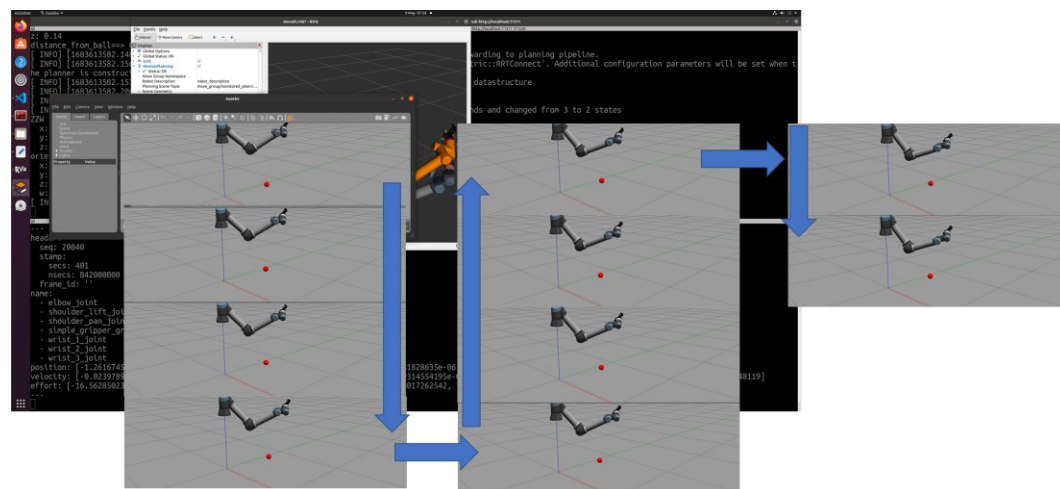


Figure 73 Training result of the RL agent reaching target.

In RL training of simplified task, the performance of RL agent may not be improved due to

various reasons. For example, the setting of reward function can be improved to make it more specific. And the criterion of the success of the task should be also well prepared in case the target is reached too soon.

The main process of RL is to train and validate the agent to fine-tune the training policy. Also, focus on the reward structure RL design policy architecture and continue the training process. RL training is time-intensive and takes minutes to days based on the end application. Thus, for a complex set of applications, faster training is achieved by using a system architecture where several CPUs, GPUs, and computing systems run in parallel.

To evaluate RL algorithms, the evaluation methods need to be improved since the inconstancy of the metrics. The inconsistency of performance stems from the use of flawed evaluation metrics. Normally, evaluation is through each episode, which contains many timesteps¹⁹¹. In this study, the average reward in a time step will be used to evaluate the performance of RL algorithms.

To save the RL model, a customised code section should be designed. Since the original `model.save` function can only save the model after the training is finished, and the training environment sometimes collapses due to heavy computing load, the model cannot be saved at the end of the training. A callback function is inserted in the model training function to save the RL model in every 100 episodes.

In this subsection, it was proved that the customised platform with ROS+ Gazebo + Moveit + OpenAI_ROS+ Stable_baselines3 can implement RL algorithms in ROS Gazebo environment. The simulation ran smoothly, and the reward can be evaluated using the tensorboard log. The simplified RL task will be extended to surface scanning in the next subsection.

5.3 Simulation model with RL algorithm

For implementing surface scanning tasks using RL, a flat surface scanning task is carried out at first. This task is to validate the basic surface scanning implementation of RL robot model. once the flat surface can be scanned by the model, the curved surface can be implemented for the next step. A table is spawned in the world at first. The target ball is located on the table surface to mimic the scanning target on the table (see Figure 74). In this scenario, the ball indicates the moving target of the end-effector. 4 actions are defined: moving forward (y++), backward (y--), left (x++) and right (x--). The delta movement distance for each time step is set to be 5cm. The acceptance distance of the task is set to be 15cm. For each movement closer to the target, the reward is 10. If the task is done, the reward is 100. All the movement of the end-effector should be attached to the table surface.

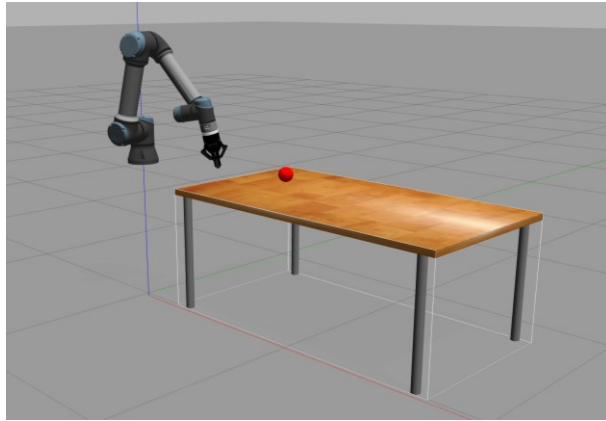


Figure 74 Simulation environment for flat surface scanning.

To move the end-effector along defined paths, the poses need to be set using code. The reason is that if the position of the end-effector is changed, the orientation needs to be changed to make the actual orientation the same as the previous one. For example, to make the end-effector normal to the ground, and move the position from (0.3944, 0.3331, 0.3765) to (0.4944, 0.3331, 0.3765), the orientation needs to be changed from (x: 0.049964109711171496, y: -0.9978167636671763, z: -0.012601549681218514, w: 0.041309742569569693) to (-0.049964109711171496, 0.9978167636671763, 0.012601549681218514, -0.041309742569569693). In the context of robot orientation, the terms x, y, z, and w often refer to the components of a quaternion, which is a mathematical representation of orientation in three-dimensional space. Quaternions are commonly used to represent rotations in robotics and computer graphics. The quaternion is usually denoted as $q = (x, y, z, w)$, as w is the scalar (real) part of the quaternion. x, y, z are the vector (imaginary) parts of the quaternion. Together, these components define the orientation of the robot in 3D space. The quaternion can be used to represent a rotation about an arbitrary axis. PyKDL movement transformation was used to set up the actions. PyKDL is used to transfer the current orientation to next pose. So the position of end-effector will be moved, but not the orientation. The function used is PyKDL. It uses a transformation matrix to transfer current pose to a KDL frame. Then, use the movement data, such as displacement in x,y,z axis and the rotation around roll, pitch, yaw directions to build a transformation matrix to change the target frame in the next step.

Due to setting up of Gazebo, the starting state of the robot is not properly prepared, and the simulation cannot continue. In this case, the joint states of the robot are set back to the initial pose and restart the trajectory. During the movement of the robotic arm, a constraint is set for the orientation of the end-effector to make the flat surface scanning task easier.

It can be seen that at the beginning of the training, the robot moves randomly. At most of the time, it moved backward and right to move farther away from the target. The reward is

definitely low, sometimes even negative.

In this study, two RL algorithms, i.e., PPO¹⁹² and A2C¹⁹³ algorithms are used to validate the proposed method. For the PPO (Proximal Policy Optimization) algorithm, it is a policy-based algorithm. A policy is a function that maps a given state of an environment to an action that the agent should take in that state. The policy defines the behaviour of the agent in the environment and determines how the agent interacts with the environment to achieve its goal. On the surface level, the difference between traditional policy gradient methods (e.g., REINFORCE) and PPO is small. In the context of RL, a policy π is simply a function that returns a feasible action at the given state s . In policy-based methods, the function (e.g., a neural network) is defined by a set of parameters θ . To identify PPO from other policy-based algorithm, the core idea is to replace the deterministic policy $\pi: s \rightarrow a$ with a parameterized probability distribution $\pi_\theta(a|s) = P(a|s; \theta)$. Instead of returning a single action, we sample actions from a probability distribution tuned by θ . These parameters can be adjusted, researchers can observe the differences in resulting rewards, and update θ in the direction that yields higher rewards. This mechanism underlies the notion of all policy gradient methods. The policy $\pi_\theta(a|s)$ is stochastic, meaning that the parameters dictate the sampling probability of actions a , and thereby influence the probability of following trajectories $\tau = s_1, a_1, \dots, s_n, a_n$. The objective function of PPO can be described as:

$$J(\theta) = E_{\tau \sim \pi_\theta} * R(\tau) = \sum_{\tau} P(\tau; \theta) * R(\tau) \quad (61)$$

in which R is the reward value, P is the probability of taking related actions.

In policy-based method, the policies are stochastic. When various actions are sampled for measuring differences in rewards with corresponding probabilities. The observed rewards, combined with the action probability, yields a reward signal. To determine the update direction that improve the objective function, policy gradient methods rely on gradients ∇_θ (a vector of derivatives). This yields the following update function:

$$\theta \leftarrow \theta + \alpha * \nabla_\theta * J(\theta) \quad (62)$$

in which α stands for learning rate, θ stands for policy weights.

The adjustment of θ has a big impact on the training result. So the approach is straightforward, by increasing the probabilities of high reward trajectories, the expected reward can be improved. Take the classic REINFORCE algorithm (or Vanilla policy gradient) as an example, it calculates the cumulated reward by trying out state-action trajectories. Then, θ can be updated in the direction that yields higher rewards.

What makes PPO different is the PPO penalty and PPO clip. During training, a “surrogate optimisation” technique to update the policy parameters. The surrogate objective function used in PPO is designed to both maximize the expected reward and constrain the policy update to be close to the previous policy. At first, PPO collects a set of trajectories by running the current policy in the environment for a fixed number of steps. For each time step, the policy outputs an action given the current observation. For each time step in each trajectory, PPO computes an estimate of the advantage function, which is the difference between the observed reward and the expected reward. This advantage estimate measures how much better or worse the actual action taken was compared to the average action taken by the current policy. Using the collected trajectories and the computed advantage estimates, PPO then calculates a surrogate objective function that is designed to maximize the expected reward and also ensure that the new policy is not too different from the previous policy. The surrogate objective function is based on the probability ratio between the new and old policies, weighted by the advantage estimate. Finally, PPO updates the policy parameters by optimizing the surrogate objective function using a stochastic gradient descent (SGD) algorithm. The update is typically done using mini batches of trajectories to improve computational efficiency. These steps are repeated for a fixed number of epochs, where each epoch consists of collecting new trajectories, computing advantages, computing the surrogate objective, and updating the policy parameters.

The advantage of PPO is that it is a good balance on efficiency and comprehension of RL, it works well in high-dimensional spaces. PPO is particularly effective in high-dimensional action spaces, such as those found in robotics and video games. It is because PPO uses a neural network to represent the policy, which can handle large state and action spaces. Moreover, PPO is relatively robust to hyperparameters, meaning it can perform well across a range of hyperparameter settings. This is because PPO uses a clipping mechanism that makes it less sensitive to hyperparameters such as the learning rate and batch size. With the penalty, PPO checks the size of the update after each update. If the realized divergence exceeds the target divergence by more than 1.5, for the next iteration we penalize divergence harder by doubling β . The other way around, we halve β if updates are too small, effectively expanding the trust region. With the clip, the range of the penalty is restricted within which the policy can change. The clipped PPO is reported to have a better result than penalty methods.

While the shortcomings of PPO are that as a policy-based method, the inconsistent policy update overshoot the results and sometimes make the agent missed the peak reward. Sample inefficiency as the samples are only used once. After that, the policy is updated and the new policy is used to sample another trajectory. As sampling is often expensive, this can be prohibitive. And as the policy gradient is a Monte Carlo learning approach, taking into account the full reward trajectory (i.e., a full episode). Such trajectories often suffer from high variance,

hampering convergence. If the reward is 0, then neither the good actions or the bad actions will not be learned.

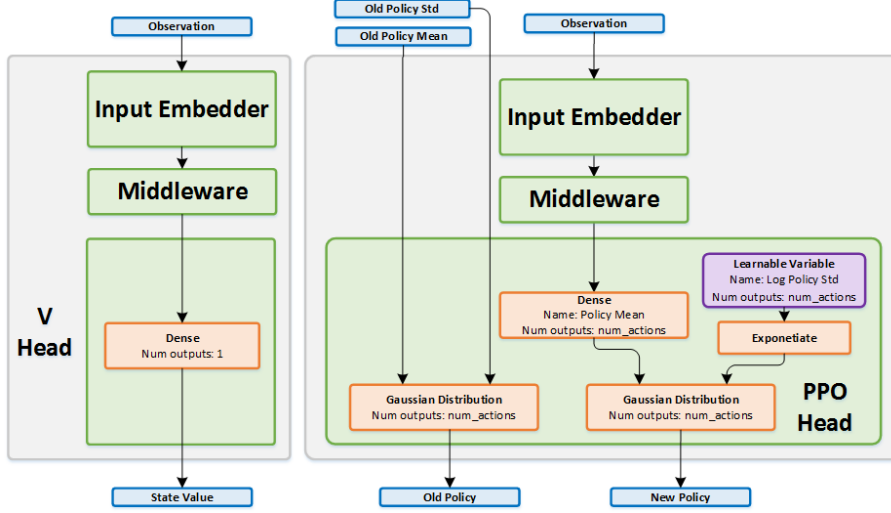


Figure 75 Diagram of PPO algorithm

(link: https://intellabs.github.io/coach/components/agents/policy_optimization/cppo.html).

To implement PPO in simulation, the algorithm can be directly used from the “stable_baselines3” library. However, to improve the functions of PPO, an improved PPO algorithm is coded and loaded in the main code of training.

A2C (advantage actor-critic) algorithm is another latest RL algorithm, which uses actor and critic to “discuss” an optimal solution. In A2C, a baseline strategy is used to tackle high deviance problem in PPO algorithm. The “Critic” estimates the value function. This could be the action-value (the Q value) $Q_w(s_t, a_t)$ or state-value (the V value) $V_v(s_t)$. The “Actor” updates the policy distribution $\pi_\theta(s, a)$ in the direction suggested by the Critic (such as with policy gradients), which makes this method like a combination of value-based and policy-based method. Both the Critic and Actor functions are parameterized with neural networks. Intuitively, the advantage value means how much better it is to take a specific action compared to the average, general action at the given state. Similar to PPO algorithm, θ indicates the parameters in policy neural network. ω stands for the parameters in value estimation neural network in critic. v is the v-function as the baseline function in estimating the V-value.

$$A(s_t, a_t) = Q_w(s_t, a_t) - V_v(s_t) \quad (63)$$

in which Q_w is the Q-value, and V_v is the V-value. The V-value stands for the average of the state, the advantage value can indicate how well the action affects the environment. And since the Q-value and V-value have relationship like this:

$$Q(s_t, a_t) = E[r_{t+1} + \gamma V_v(s_{t+1})] \quad (64)$$

At each timestep, t , we get the current state s_t from the environment and pass it as input through our Actor and Critic. Our Policy takes the state and outputs an action a_t . After the acquisition of the state and action, the Q-value can be calculated. Therefore, once (4) and (5) are combined, only one neural network is needed to calculate optimal solutions, and this is the advantage that A2C has over value-based AC methods. The descriptive model of A2C algorithm is shown in Figure 76.

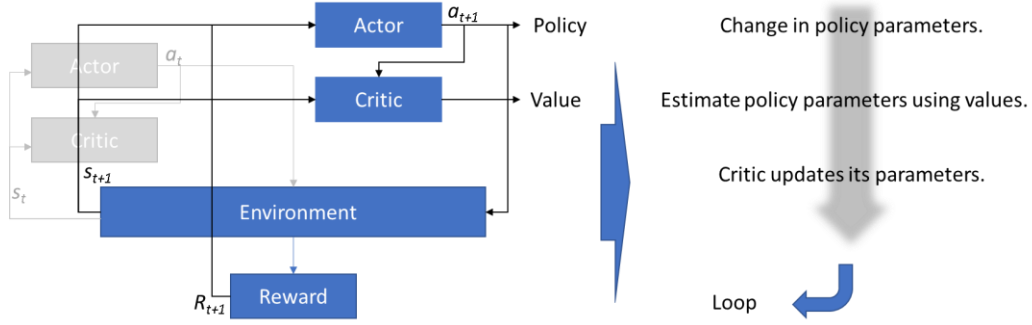


Figure 76 Diagram of A2C algorithm.

$$\Delta\theta = \alpha \nabla_{\theta}(\log \pi_{\theta}(s, a)) \hat{q}_w(s, a) \quad (65)$$

$$\Delta\omega = \beta [R(s, a) + \gamma \hat{q}_w(s_{t+1}, a_{t+1}) - \hat{q}_w(s_t, a_t)] \nabla_{\omega} \hat{q}_w(s_t, a_t) \quad (66)$$

In (65), $\Delta\theta$ is the change in policy parameters (weights). α is the learning rate. ∇_{θ} is the gradient of θ function. $\nabla_{\theta}(\log \pi_{\theta}(s, a))$ indicates the gradient of the function $\log \pi_{\theta}(s, a)$ with the respect to θ . In (66), $\nabla_{\omega} \hat{q}_w(s_t, a_t)$ is the gradient of value function. β is the learning rate. The formula in the square bracket is the TD error as a good estimator of the advantage function (comes from (63)).

To compare the algorithms, a simulation test program is designed, to validate the current algorithm with different parameters (see Table 8).

Table 8 The criteria parameters of RL simulation comparison evaluation.

Item	Details	
RL algorithm used	<i>PPO</i>	<i>A2C</i>
Time steps in each episode	<i>100</i>	<i>1000</i>
Delta movement in each action	<i>5cm</i>	<i>10cm</i>
Acceptance distance for reward	<i>5cm</i>	<i>10cm</i>

For the long-horizon RL training tasks, there have been problems that block the efficiency of learning. The first is the sparse reward given and the other is the adaptability of the trained model. a slight change in the environment can affect the performance in practice hugely. To

overcome these two problems, a new proposed RL method is carried out in this study.

At first, the whole task is divided into a sequence of sub-tasks. Reward is defined for each sub-task. For long-horizon tasks, if the reward can be only achieved when the whole task is finished, immature policy finished in earlier stage will be taken. In this study, good and bad actions are all assigned a reward. Positive reward for good actions and negative reward for bad actions. This arrangement is to balance the “exploration and exploitation” problem in RL. In the definition of training processes, the pose of the robot can be only reset to the original pose when the robot is stuck, or the joint movement is out of limits. In this case, the robotic arm can not only learn from well-rewarded actions, but also from the negative-rewarded actions. In the “ball-reaching” task, in each action, if the end-effector is moving closer to the target, the reward will gain 10. The reward will reduce 10 if it moves farther from the target. If the end-effector reaches the target, the reward will gain 100. In the reaching task, the delta movement of the end-effector is pre-defined, however, due to simulation environment and reverse-kinematic reasons, sometimes the movement of the end-effector is less than the delta value. In this case, the robot is set back to the original pose. And the reward calculation will continue. The timer of the Gazebo simulation should be restarted in code. For the adaptability, transfer learning is proposed for the implementation in real world, which will be introduced in the next section.

Another problem for RL is that the exploration of actions is too slow at the beginning of the training. By comparing different setups, it can be seen that if the limits and constraints of kinematics parameters of robotic arm is well designed. And the action space, within which the robotic arm operates, is very important to the training. If the action space can be detailed defined, the training will focus more on the assigned space and converge quicker. The detailed action space can also prevent the joint problem during actions, e.g., joint position moves out of range.

The results of RL training are listed in Table 9. The best reward and the time steps to achieve it are listed as the assessment.

It can be seen from the results that the A2C is better for the task, since it took shorter time to reach the terminal of training. And as expected, as the acceptance distance is bigger, the training time is shorter. And as the timesteps get more in each episode, the model tends to find better solutions for the task.

Experiment 5(E5): comparing two RL algorithms with the proposed ROS platform. To compare the actual performance of GPU with the CPU, a PC is used to test the same algorithm with GPU and CPU. The setup of PC is: Intel Core i5-12400F processor, NVIDIA GTX3060 GPU, 16Gb memory. Testing the same A2C algorithm with 100 time episodes, CPU outperformed GPU, 30,161,925.2018ms vs. 34,238,581.8924ms. It can be shown that CPU performed close to GPU, the possible reason is that the A2C algorithm was not well-tuned for GPU performance. The complexity of neural networks in A2C is not sufficient to unleash all the computational power of GPU. Another possible reason for this may due to the Bottleneck in GPU Usage. GPUs have a limited number of cores, and if the A2C algorithm does not fully utilize these cores or is not efficiently parallelized, the GPU may not demonstrate its full potential. This comparison test will be discussed in the later part of this thesis.

To evaluate the effectiveness of algorithms, another parameter, success rate is introduced to the results. For example, if you have 1000 time-steps and the robot successfully completes the task in 800 of those time steps, the success rate would be 80%. To implement this definition, you can track the successful completion of the manipulation task within the time step and increment a success counter accordingly. At the end of the simulation or at specific intervals, you can calculate the success rate by dividing the number of successful time steps by the total number of time steps.

To make the scanning task more complete, contact force measurement is included during the scanning. In flat surface scanning task, since the surface is flat, the orientation optimisation is not implemented. So the next step is to add the orientation optimisation

While the settings of robot, training parameters can solve the convergence problem and the exploration and exploitation problem, the RL network parameters should be also studied to

Table 9 The results of simulation tests with A2C algorithm.

	100 step/5/5	100 step/5/10	100 step/10/5	100 step/10/10
PPO	80(reward)/100(steps) (9.98 hours)	150/95 (9.54h)	80/100 (9.55h)	180/90 (9.53h)
A2C	100/80 (7.38 hours)	160/75 (7.23h)	80/60 (7.17h)	190/70 (7.3h)
	1000 step/5/5	1000 step/5/10	1000 step/10/5	1000 step/10/10
PPO	500/100 (106.30 hours)	580/100 (100.30 hours)	500/100 (99.54 hours)	600/90 (98.94 hours)
A2C	600/80 (78.90 hours)	650/75 (76.43 hours)	550/75 (75.90 hours)	680/75 (75.54 hours)

improve the efficiency and effectiveness of training. Since A2C is a model-free algorithm to tackle RL problem also with modelling the policy, it is more complex to build the algorithm and it is more sensitive to the option of hyperparameters, such as learning rate and neural network size.

The A2C algorithm uses a shared neural network architecture for both the actor and critic components. It consists of layers that process the input observations and output the action probabilities and value estimates. The neural network takes the environment observations as input, which is the position and pose of the end-effector in this study. The initial layers of the neural network are typically shared between the actor and critic components. These layers capture common features and representations from the observations. The actor head is responsible for outputting the action probabilities. It typically consists of one or more fully connected layers followed by a softmax activation function to generate a probability distribution over the available actions. The critic head estimates the value function, which represents the expected cumulative reward from a given state. It usually consists of one or more fully connected layers that output a single value estimate. The activation functions used in the neural network layers can vary, but commonly used options include ReLU (Rectified Linear Unit) or tanh (hyperbolic tangent) activation functions. The A2C algorithm uses appropriate loss functions for both the actor and critic components. The actor loss is often defined using the advantage estimation, while the critic loss is typically based on the mean squared error between the estimated values and actual returns.

In the source code of A2C in `Stable_baselines3` library, there are some hyperparameters that can be modified to improve the performance. `n_steps`: The number of steps to collect experience before performing a gradient update. This parameter affects the trade-off between exploration and exploitation. Increasing `n_steps` can lead to more stable updates but may slow down learning. The discount factor, `gamma`, for future rewards. It determines the importance of future rewards compared to immediate rewards. Adjusting `gamma` can influence the agent's preference for immediate rewards versus long-term planning. `Lambda` is the Generalized Advantage Estimation (GAE) parameter. It controls the trade-off between bias and variance in estimating the advantages. A higher value of `gae_lambda` increases bias but reduces variance. The learning rate for the optimizer used in the A2C algorithm. It controls the step size during parameter updates. Modifying the `learning_rate` can impact the speed and stability of learning. `ent_coef`: The coefficient for the entropy regularization term in the A2C objective function. It balances exploration and exploitation by encouraging exploration through increased policy entropy. Adjusting `ent_coef` can affect the level of exploration the agent exhibits. `vf_coef`: The coefficient for the value function loss term in the A2C objective function. It determines the importance of the value function estimation during optimization. Modifying `vf_coef` can

influence the agent's focus on value estimation versus policy improvement.

The neural network of A2C algorithm is to estimate both the policy and the value function of the algorithm. After the input of environmental values, two values, i.e., policy and value, are output. Combining policy and value, the policy will guide agent to take better actions, while the value provides the advantages of current action. So, the policy estimator acts like “actor”. The policy network, implemented as part of the actor, helps the agent make decisions on which action to take given the current state. The output of the policy network is a probability distribution that assigns probabilities to each possible action, indicating the likelihood of selecting each action. The value estimation is acting as the “critic” role. During training, the critic's role is to help the agent learn more accurately assess the value of different states. By estimating the state values, the critic provides a baseline for the advantage estimation, which is the difference between the observed rewards and the expected values. The advantage is then used to update the policy and guide the agent's actions towards more advantageous states. By customise the “MlpPolicy” function, it can be modified and improve the performance.

To customise the neural network, the modification from the “MlpPolicy” class can be implemented. Inside the CustomMlpPolicy class, you can modify the `net_arch` parameter to define your desired network architecture. In this case, the network architecture is set to have two hidden layers with 256 and 128 units, respectively. The “MlpPolicy” represents a Multi-Layer Perceptron (MLP) neural network-based policy. It is a simple and widely used policy architecture for reinforcement learning algorithms.

The MlpPolicy consists of a feedforward neural network with multiple fully connected layers. The number of layers and the number of units in each layer can be customized based on the problem requirements. By default, the MlpPolicy uses the Rectified Linear Unit (ReLU) activation function for hidden layers. ReLU helps introduce non-linearity into the network, allowing it to learn complex patterns and representations. The MlpPolicy has separate output layers for the policy and value function estimation. The policy output layer typically uses a softmax activation function to produce a probability distribution over available actions, while the value output layer provides a single value estimation for the state value. The MlpPolicy allows for customization of the network architecture through the `net_arch` parameter. It accepts a list of integers, where each integer represents the number of units in a hidden layer. By modifying this parameter, you can easily adjust the number of hidden layers and the number of units in each layer. The MlpPolicy is compatible with various reinforcement learning algorithms in the `Stable_Baselines3` library, including A2C, PPO (Proximal Policy Optimization), and SAC (Soft Actor-Critic). Normally, the actor and critic use separate neural networks. In `stable_baselines3` library, the same network structure is used for the actor and

critic before the output layer. This makes it easy to switch between different algorithms while using the same policy architecture (see Figure 77).

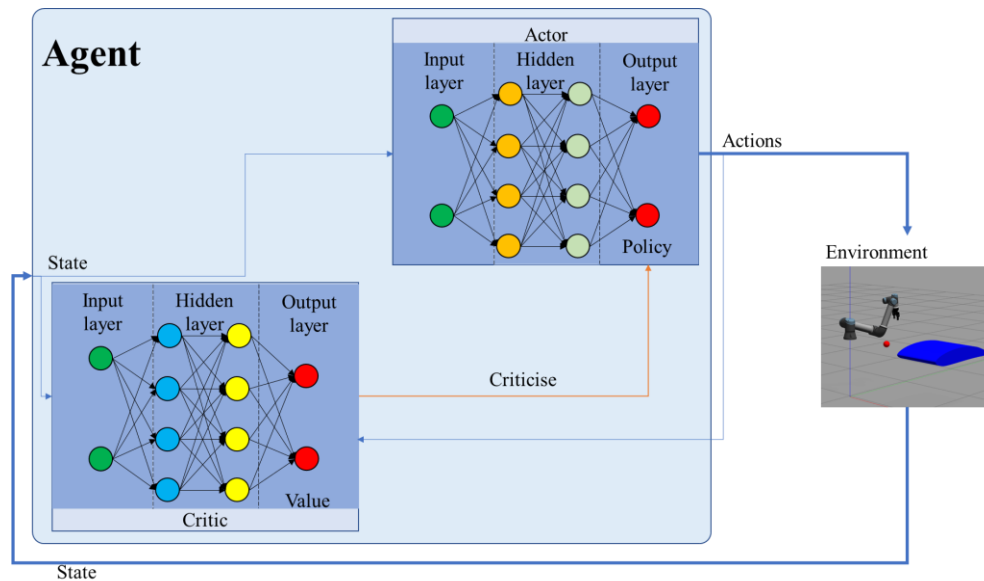


Figure 77 Diagram of neural network in A2C

The neural network in A2C is more of more traditional fully connected networks, i.e., multi-layer perceptron (MLP), instead of convolutional neural network (CNN). The reason is the NN in A2C is not for classification, and the output of MlpPolicy function is divided into two branches, actor and critic. MLPs, also known as fully connected networks, consist of multiple layers of neurons where each neuron is connected to every neuron in the adjacent layers. MLPs lack spatial or temporal structure as they don't leverage convolution or recurrence. MLPs are commonly used in A2C when the observations are represented as a flattened feature vector or when there is no inherent grid or temporal structure in the data. MLPs are versatile and can model complex relationships in high-dimensional feature spaces. If the task is image-related, CNN policy neural network can be used to capture spatial features and patterns.

For the input layer, the function is to transfer the observation information into RL training. A linear transformation (or fully connected) layer is normally used. It is used to transfer the dimensionality of the observation into the dimension that the hidden layer needs. The input data, which could be a vector or a set of features are fed into the input layer, where a linear transformation is applied. This transformation involves a matrix multiplication and a bias addition. The output of the input layer is the transformed representation of the input features. This representation is passed to the subsequent layers of the neural network for further processing and learning. Although there are other kinds of functions, the input layer is limited to linear transformation or none activation.

After the input layer, an activation function is used to pass through the input to hidden layer.

The ReLU activation function introduces non-linearity into the network, enabling it to learn complex patterns and relationships in the data.

ReLU is the most common hidden layer function, within ReLU, it introduces non-linearity to the network by mapping negative values to zero and keeping positive values unchanged. $\text{ReLU}(x) = \max(0, x)$. By setting negative values to zero, ReLU effectively removes the negative saturation, allowing the network to focus on important and relevant features in the data. The primary motivation behind using ReLU is to address the vanishing gradient problem. In deep neural networks, gradients can diminish as they propagate backward through layers, making it difficult for earlier layers to learn effectively. ReLU helps alleviate this problem by maintaining non-zero gradients for positive inputs.

In the context of neural networks, the input values can be negative depending on the weights and biases associated with the connections between neurons. The negative input values can arise due to various factors, such as the initial weights, the specific data being processed, or the learned weights during training. It's important to note that negative inputs are not inherently bad or undesirable in neural networks. In fact, negative input values contain valuable information and contribute to the overall learning process. Neural networks are designed to handle both positive and negative inputs to capture complex patterns and relationships in the data.

In the context of the A2C RL algorithm, the ReLU activation function is typically applied to learn complex relationships and patterns. The role of ReLU in the A2C algorithm is to capture non-linear dependencies between the input features. By setting negative values to zero, ReLU effectively removes the negative saturation and introduces sparsity in the network, which can be beneficial for learning more expressive and efficient representations. ReLU has become a popular choice for activation functions in neural networks due to its simplicity and effectiveness. It avoids the vanishing gradient problem, which can occur with activation functions like sigmoid and tanh, by allowing for more efficient gradient flow during backpropagation. Additionally, ReLU can help neural networks learn faster and converge more quickly compared to other activation functions. In the A2C algorithm, the Actor and Critic components share the same neural network architecture, including the ReLU activation function. The shared MLP policy network with ReLU activation is responsible for generating action probabilities (Actor) and estimating state values (Critic). By applying ReLU activation after the linear transformations in the hidden layers, the network can learn to model non-linear relationships in the environment and improve the policy and value estimates. However, when the input has too many negative bias, ReLU is not functioning, various modifications and variants of ReLU, such as Leaky ReLU and Parametric ReLU, have been proposed to mitigate this issue.

Other than ReLU, Sigmoid, Tanh, softplus, ELU, etc algorithm have been used in hidden layer for other applications. These are just a few examples of activation functions that can be used in fully connected layers of the A2C algorithm. The choice of activation function depends on the specific problem, network architecture, and desired behavior of the network. Experimentation and tuning are often required to find the most suitable activation function for a particular task.

Since A2C algorithm will generate two outputs, i.e., policy and value, the output layer is different for actor and critic. A softmax function is often used in actor case since it is designed for multi-class classification problems. The softmax function takes a vector of input values and produces a probability distribution over multiple classes. It exponentiates the input values and normalizes them so that the resulting values sum up to 1. Each output value represents the probability of the corresponding class. In the context of A2C, the softmax layer is typically used in the Actor component of the algorithm. The Actor is responsible for generating action probabilities, and the softmax function is applied to the output of the preceding hidden layer to convert the values into a probability distribution over possible actions. For the Critic component of A2C, which estimates state values, does not typically involve a softmax layer in its output. The Critic may use a linear activation or another appropriate activation function depending on the specific requirements of the task.

The number of epochs affects how many times the algorithm performs gradient updates and policy improvements based on the collected data. The number of epochs is considered a hyperparameter. It does not directly correspond to the total number of timesteps or interactions with the environment. It's important to note that the interpretation and usage of epochs may vary across different reinforcement learning libraries and algorithms. Therefore, it's always recommended to consult the specific documentation or source code of the algorithm you are using for accurate details on how epochs are defined and utilized. Learning algorithms take hundreds or thousands of epochs to minimize the error in the model to the greatest extent possible. The number of epochs may be as low as ten or high as 1000 and more.

Experiment 6(E6): tuning hyper-parameters for A2C algorithm. To improve the performance of A2C, neural network structure is also tuned in this study. Learning rate indicates the extent to which an agent updates its estimated values or policies based on new experiences or feedback. It is a crucial hyperparameter that influences the speed and stability of learning. The exploration/exploitation rate is set to be higher than normal, i.e., 0.1 as in the Epsilon-Greedy exploration. since RL has the exploration rate, during training, some actions beyond the normal action space can be seen. Except for exploration rate, other factors, such as optimiser type, learning rate, can be tuned in NN. The hyperparameters of RL training are compared in Table 10.

Table 10 The comparison of results with different hyperparameters in A2C algorithm.

	Value	Results	Value	Results
Optimiser	RMSprop	550/100	Adam	570/100
Learning rate	0.0007	580/90	0.001	620/70
Discount factor	0.99	550/100	0.95	570/100
Value function coefficient	0.5	560/100	0.8	550/100

From the training results, it can be shown that Adam optimizer can achieve more reward. When the learning rate is bigger, the RL can reach higher reward faster. To compare the results better, the 0.0001 learning rate was also tested. The time to reach 180 reward was 10 times longer than 0.001 learning rate. Moreover, the bigger learning rate has the bigger success rate, the 0.001 learning rate has a 2 times bigger success rate than original learning rate. Comparing to 0.99, the 0.95 discount factor can achieve better reward. The value function coefficient 0.5 and 0.8 have similar rewards, so the original value 0.5 was chosen. As a conclusion, Adam optimizer, 0.001 learning rate, 0.95 discount factor, and 0.5 value function coefficient are chosen as hyperparameters of A2C RL training.

A smaller discount factor in reinforcement learning algorithms, you can set the discount factor (also known as the gamma value) to a value less than the default value of 0.99. The discount factor determines the importance of future rewards compared to immediate rewards in the reinforcement learning process. A smaller discount factor places less emphasis on future rewards and may result in the agent focusing more on immediate rewards. For example, in the result, it can be seen that the agent reward hit 160 in 37 timesteps, while with original discount factor, 11 steps were taken. It can be noted that with smaller discount factor, the success rate was smaller than original factor, 1.8% vs 2.2%. The variance of original factor is 28% bigger than the gamma 0.95 rewards. It means, that the bigger discount factor focuses more on the overall improvement of rewards.

In Stable Baselines3, the default value function coefficient for the Advantage Actor-Critic (A2C) algorithm is 0.5. This coefficient determines the weight given to the value function loss compared to the policy loss during the training process. In A2C, the total loss consists of two components: the policy loss and the value function loss. The policy loss is responsible for updating the policy to maximize the expected cumulative reward, while the value function loss helps to improve the estimation of the state-value function. The default value function coefficient of 0.5 means that the value function loss is weighted equally with the policy loss.

However, this value can be modified if desired, allowing you to give more or less importance to the value function during training.

In reinforcement learning tasks for robots, various result parameters are commonly discussed to evaluate the performance and effectiveness of the learning algorithms. These parameters provide insights into the robot's behavior, learning progress, and achievement of task objectives. Some of the commonly discussed result parameters in reinforcement learning for robots include:

Reward: The reward received by the robot during the learning process. The reward signal is an important component of reinforcement learning, guiding the robot's actions towards maximizing cumulative rewards.

Cumulative Reward: The total sum of rewards accumulated by the robot over an episode or a series of episodes. It indicates the overall success or performance of the learning algorithm in achieving the task objectives.

Episode Length: The number of time steps or actions taken by the robot within a single episode. Episode length can provide insights into the efficiency and speed of the learning algorithm in completing tasks. The episode length should reduce when the agent trains more which means it survives easier than before.

Success Rate: The percentage of episodes or trials in which the robot successfully achieves the desired task objectives. It measures the overall success of the learning algorithm in accomplishing the assigned tasks.

Convergence: The convergence refers to the point at which the learning algorithm reaches stable and consistent behavior. It indicates that the robot has learned an effective policy and is performing optimally.

Exploration vs. Exploitation Trade-off: Reinforcement learning algorithms often balance exploration (trying out different actions to learn more about the environment) and exploitation (using the learned policy to maximize rewards). Analyzing this trade-off can provide insights into the learning algorithm's ability to explore the environment effectively while exploiting the learned knowledge.

Learning Curve: The learning curve illustrates the learning progress of the robot over time or the number of episodes. It shows how the performance (e.g., cumulative reward, success rate) evolves as the robot gains experience through learning.

Generalization: The ability of the learned policy to generalize to new and unseen scenarios or variations of the task. Evaluating the robot's performance on unseen situations measures the degree of generalization achieved by the learning algorithm.

These result parameters can vary depending on the specific task, environment, and learning algorithm being used. It is essential to define appropriate evaluation metrics that align with the objectives and requirements of the reinforcement learning task for robots.

To improve the performance of surface scanning on flat surface, force compliance controller is applied in surface scanning. After the compliance controller added to Gazebo, the reward function of RL is modified. Shown in Figure 78, the contact force during simulation can be maintained as -50N. The curve in the figure shows the force_z during scanning. The maximum error is 0.619N; the SD is 0.0285, which is stable in a simulation scanning task like this.

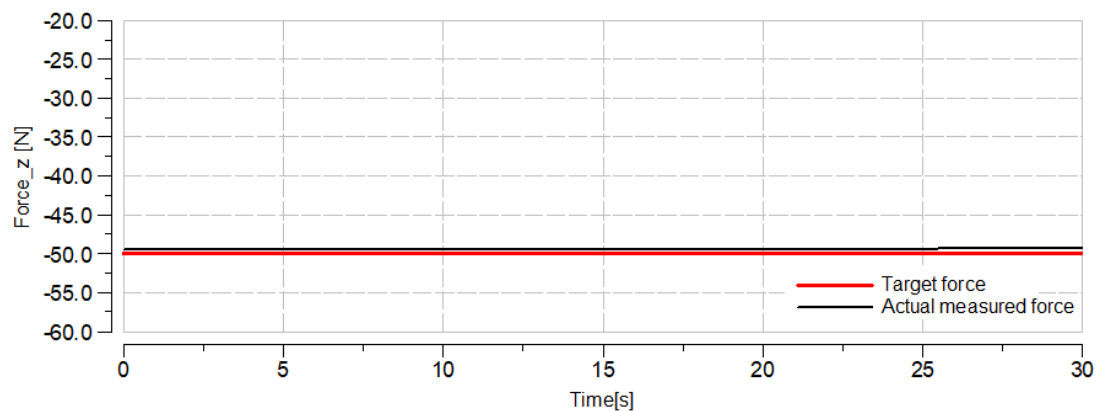


Figure 78 Force curve during one time-step in simulation.

To evaluate the force controller, the comparison between the compliance controller and a traditional proportional controller is implemented. As a traditional proportional controller, the z-axis position of the end-effector is modified as the z-direction contact force varies. To realise the final purpose of this study, an intermittent simulation was implemented firstly. The intermittent simulation is to test the workflow of RL+compliance controller on surface scanning. The target is to reach the target on the table using surface scanning. The target is a fixed ball, which is fixed on the table. The UR5e starts from its original position and pose, then the end-effector will firstly move to be normal to the table surface, then use force controller to drive the end-effector to the surface, when the contact force is larger than 5N, which means the end-effector is contacting the table surface, the compliance controller will step in. During these processes, the target wrench, which is the target of applied force is always -50N (in Gazebo environment, the measured force from the end-effector is minus when they are contacting).

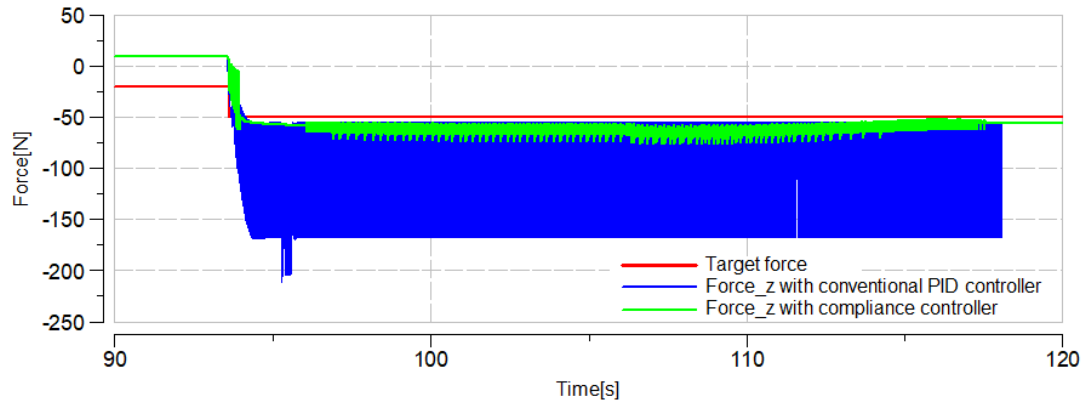


Figure 79 Force curve comparison between compliance controller vs. simple proportional controller.

It can be seen from Figure 79 that the conventional PID controller cannot realise the force control and position control task. The average of the compliance controller is -52.438N, while the PID controller average is -101N with a significant oscillation of ± 50 N. The SD of the compliance controller is 5.2338, which is much better than the PID's 56.0597. The reason for that is the PID controller is a simple controller, which makes it difficult to control a complex robotic arm system, while the compliance controller has a model in it that helps predict compliance.

Experiment 7(E7): simulation on a curved surface and implementation of RL training. To train the model in a more complex environment, a curved plane wing model is imported into the Gazebo simulation environment. The plane wing *.stl* model was established in SolidWorks. The model document was positioned in the Gazebo model library, a *.sdf* file was edited using the directory of the model file, the position and the property, including material and texture of the model (shown in Figure 80). As the RL training uses much computing resource, the redundant objects in the simulation environment, such as the table, were removed from the environment. The height of base of UR5e is 0.7m. The height of the plane wing is 0.6m. The state of plane wing is set to be static, so it will not fall on the floor. The location of the airplane wing on the table is shown in Figure 49.

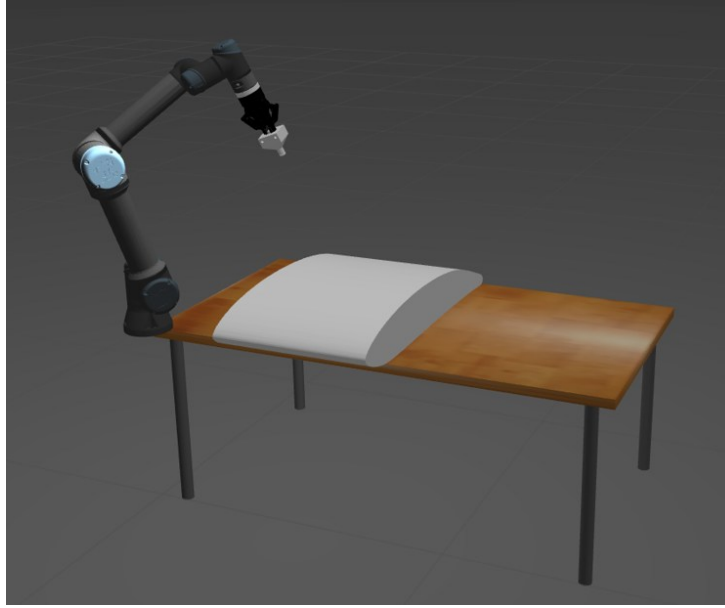


Figure 80 Simulation environment for surface scanning using RL.

As the airplane wing is the object, it is assumed that the information of the object is known in order to evaluate the RL training function, especially the control of orientation of the end-effector. As shown in Figure 81, the interpolation algorithm was used in the code to calculate the normal vector of each contact point on the surface. During the scanning process, the real-time location of the end-effector will be extracted, the actual normal vector of the airplane wing will be calculated using interpolation algorithm and compared with the actual orientation of end-effector.

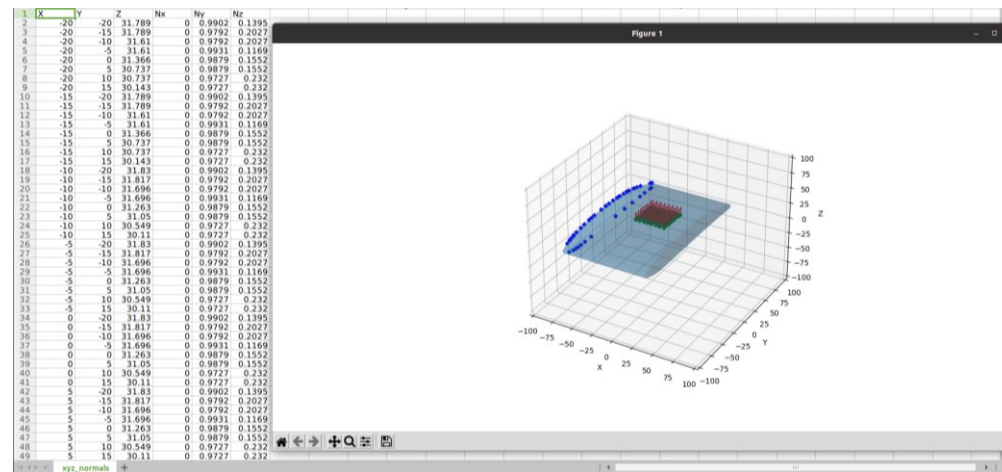


Figure 81 Model of airplane wing and the normal orientation vectors.

To optimise the orientation adjustment and to navigate the inherent uncertainties in real world, fuzzy controller is combined with RL to adjust the orientation¹⁹⁴. The fuzzy controller will output a factor, to adapt the action selection to adjust the dynamic orientation adjustment.

The reason to add fuzzy controller in the robot control is that at the beginning of each time step,

the controller will calculate the distance between the target position and the current position of the end-effector. If the end-effector has not reached the target, it will continue moving. The movement will start from current position and the action will be selected from the action space. Once the action is selected, the end-effector will move according to the action. During the action, the trajectory will be monitored, according to the data during the trajectory, the reward will be calculated, and the next step will be determined. So, the action selection will affect the trajectory and will affect the reward calculation. Even though the trajectory movement is not big, but if the trajectory is not selected carefully, it will end up with a bad end position and pose. So, it is reasonable to implement a fuzzy controller at the action selection. The trajectory correction factor can be defined in the fuzzy controller and the factor can be selected by RL and output a detailed output to correct the action to make the position better than normal trajectory (see Figure 82).

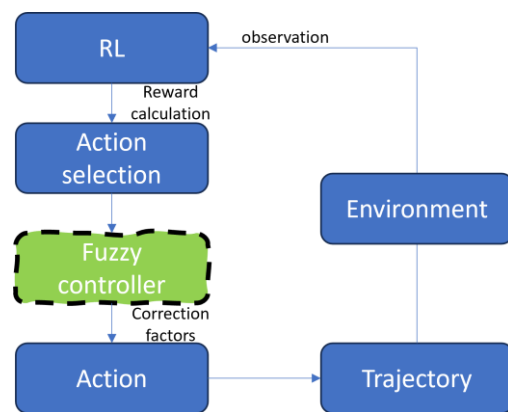


Figure 82 Fuzzy in RL controller.

Fuzzy logic controller is used for decision making and control. It has fuzzy input fuzzification model and defuzzification model. As the input values (crisp input value) are fed to the controller, it brings the value to fuzzy input model to confirm which catalog it belongs. For example, in Figure 83, there are 3 catalogues of input(1,2,3). In the membership function of the input, the crisp input value is mapped to the level of truth value. Then, the catalog will continue to operate the fuzzy rules. Within fuzzy rules, the catalog will be related to the control output categories by rules. For example, the output 1,2,3, are assigned to input 1,2,3 in Figure 83. With the level of the catalogs of input (truth value), the area of the output value will be decided. The final output value of control parameter will be decided by calculating the centroid of the total area of control value. The output value is the crisp output value(as shown in see Figure 83).

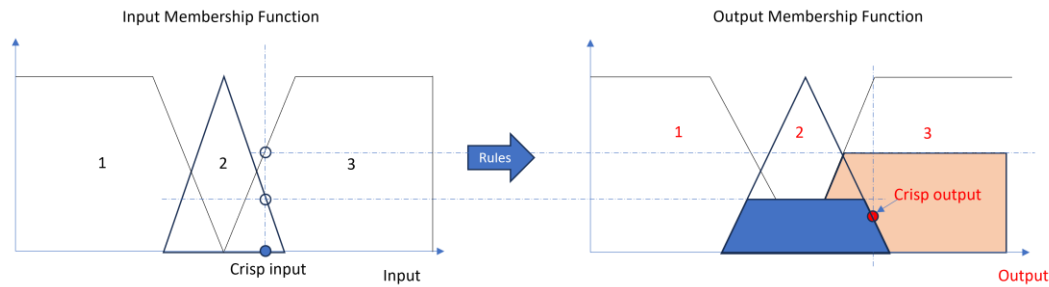


Figure 83 Fuzzy controller.

In the task of curved surface (shown in Figure 84), several factors are considered in the reward evaluation:

1. Average contact force. In the surface scanning task, the target contact force is set to -50N for the compliance controller. A data logger is set in the python code to log the actual measured contact force and calculate the difference between the actual and the target force value. The average will be calculated in every time step, if the average of the difference is within 5N, reward +10, if not, reward -10. Variance is also considered.
2. Average orientation difference, variance of orientation difference. Since the curved surface of the airplane wing is assumed to be known, the normal vector orientation can be calculated using interpolation method, in this case, KNN(k-Nearest Neighbors) algorithm. After this, the actual position of the end-effector in every time step is logged and the related actual normal vector on the surface will be calculated. A data logger is also set in the python code to log the actual end-effector orientation and the real normal vector of the curved surface as the target orientation. In every time step, the real and target values are logged and the average value and variance value of the difference are logged to evaluate the reward. Since the average value indicates the control of the orientation, and the variance value indicates the stability of the controller.
3. Distance to target. As before, the distance to the target is the main criterion of the reward. If it touches the target, reward +100. If it moves closer to the target, reward +10, if it moves farther from the target, reward -10.

As shown in Figure 85, the cumulated average of the reward of RL training increased from 137.81 to 231.71 when the time step increased. It proves that RL training can be implemented in the curved surface environment, and the scanning of the curved surface is improving during the training stage. The red line in the figure is the 3rd order polyfit of the reward data, which shows the increasing trend.

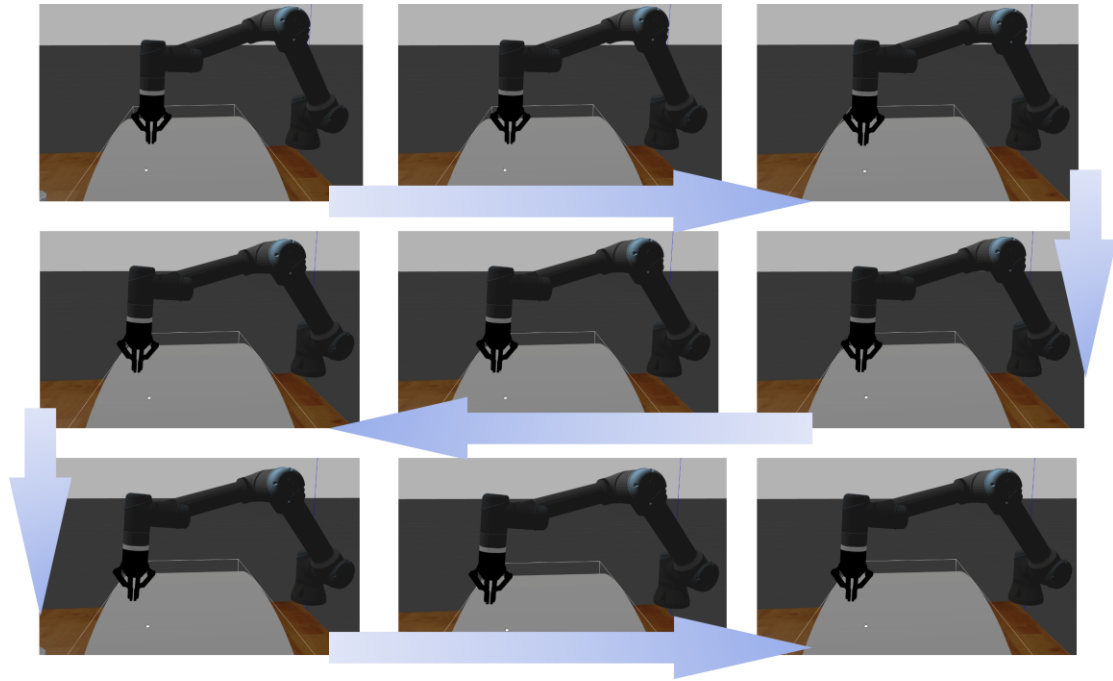


Figure 84 Training processes for curved surface.

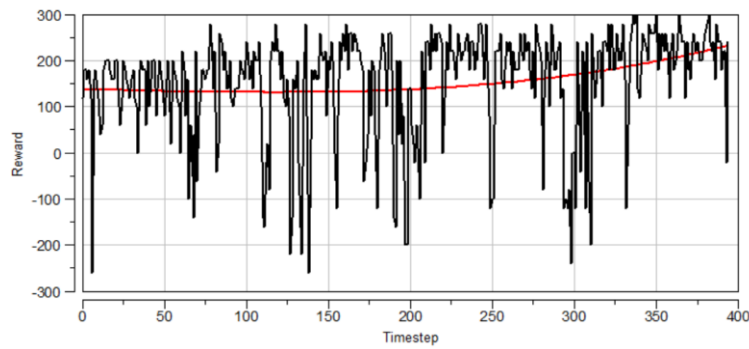


Figure 85 Reward curve of RL training on curved surface.

As also can be seen in Figure 86, in each episode, the average values of orientation difference have been recorded. As can be seen, the difference of the difference is also going down from 14.16 to 8.51. The red curve is the 3rd order polyfit curve of the data.

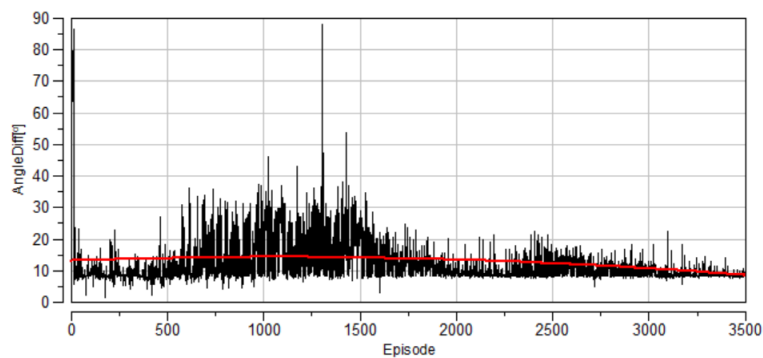


Figure 86 Orientation difference between actual and target.

For the training, it took 10.25 hours to finish the training of 200 time steps. The reward curve is shown in Figure 87.

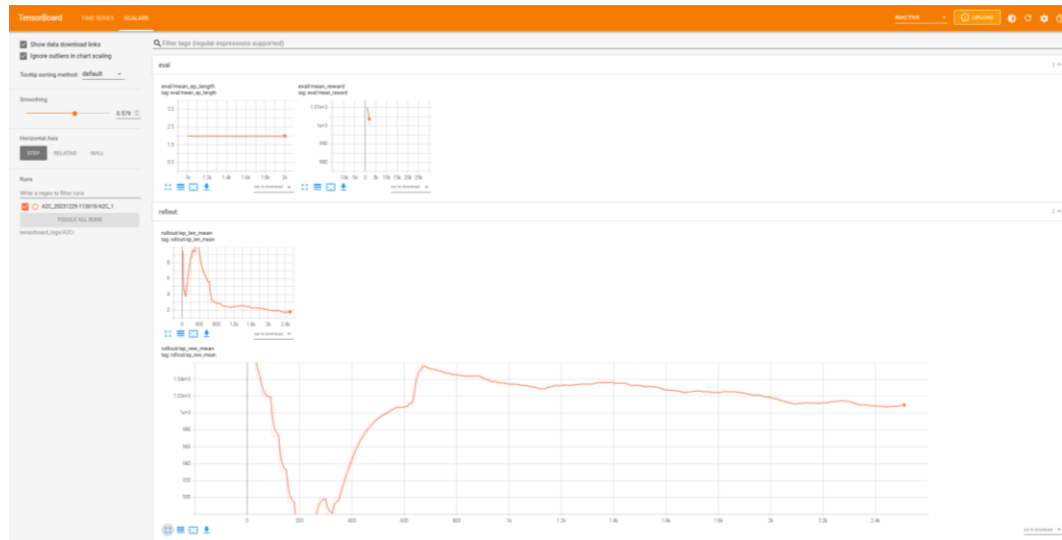


Figure 87 Screenshot of the tensorboard evaluation of curved surface RL training with “done_reward=1000”.

As can be seen in the figure, the length of the episodes dropped from the original 16 steps to later 2 or 3 steps each episode. It shows the agent has survived the environment. The original reward is good enough, 1280, however, it costed 16 steps to get there. After some episodes of training, the reward was increasing. The optimal reward for the simplified curved surface task should be 1400. It can be seen that the agent was approaching the target with less steps in the later training session. After several training sessions, it can be seen that the reward distribution may have some problem. The agent was training to achieve the target, but not getting more reward. The total reward is around 1000 and not improving, therefore, the reward strategy needs improvement. Moreover, there is a problem that the average reward is going down after the reward reaches its peak. As can be seen in Figure 87.

The reward can be re-arranged like this: the done_reward is 100, and each step that get closer to the target, force average, force variance, and angle difference can all get 20 points. Therefore, each timestep, the moving reward can get maximum 80, which is closer to the done_reward. The training scenario can be seen in Figure 88.

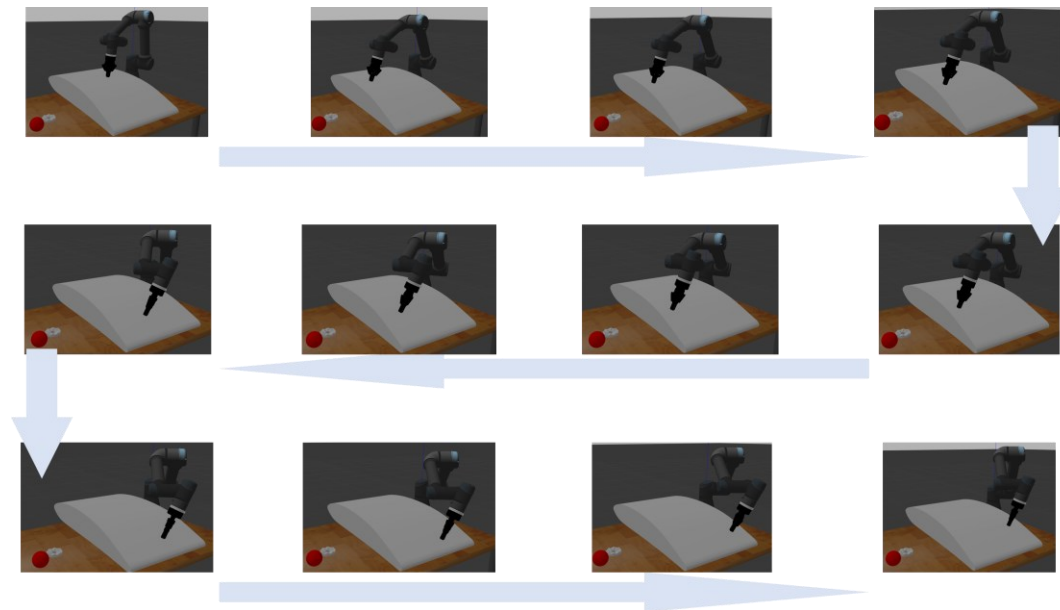


Figure 88 RL+ compliance controller+ raster scanning on curved object in Gazebo.

Subsequently, the best trained model will be used to implement the scanning task. As the orientation of end-effector is more important in the task, the orientation difference between the actual end-effector and the surface normal vector is shown in Figure 89. The results are based on the best RL trained model with the compliance controller.

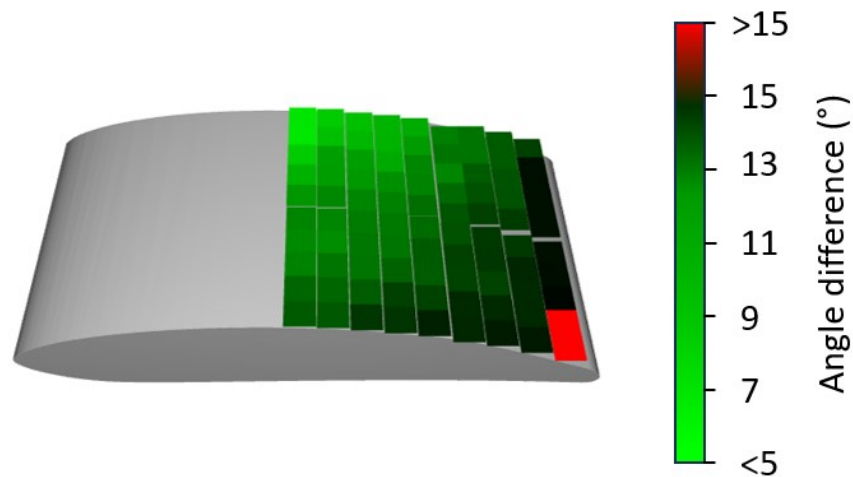


Figure 89 Accuracy of RL+compliance controller on curved wing scanning.

To compare with normal control without RL training, the normal compliance controller without any training was also carried out. The scanning results without RL can be seen in Figure 90.

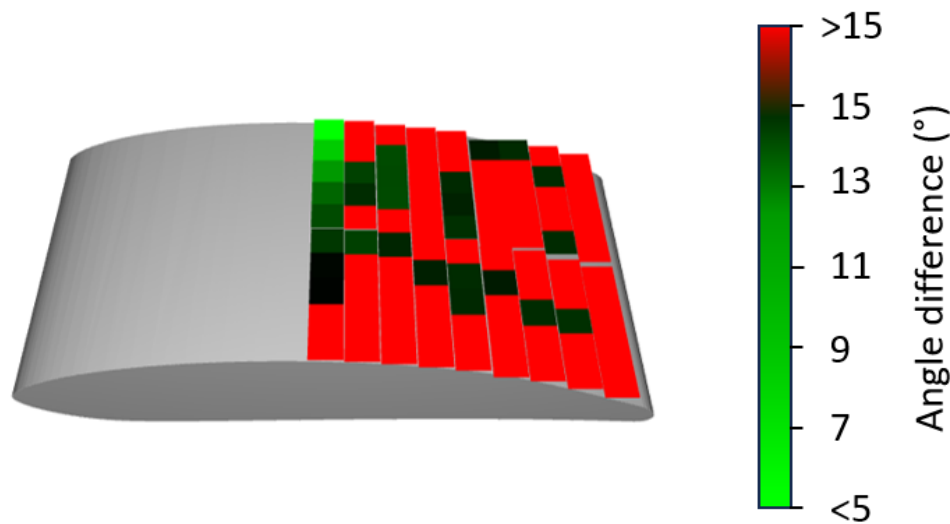


Figure 90 Accuracy of normal compliance controller on curved wing scanning.

It can be seen from the figures that with RL + compliance controller, the orientation performance is better than only compliance controller. Moreover, it can be seen that the farther the end-effector can get, the bigger difference between the actual end-effector and the normal vector of the surface will be. This is decided by the feature of the compliance controller which also depends on the torque supplied by the joint. When the end-effector reached the farthest location, it is close to the singularity position, which cannot support orientation control and compliance control. The average force value of non-RL controller is -78.325N which is 42.055% larger than RL trained model (55.137N). And it can be seen from Figure 89 and Figure 90 that the non-RL trained model is overall 30% worse in the orientation performance, based on the angle difference data.

Experiment 8(E8): simulation on the target object. After the training of curved surface, the real scanning scenario is implemented in the environment. Firstly, the simulation model of robot should be added with the ultrasonic transducer holder. Subsequently, it should scan on the target object, the defected die (as shown in Figure 91).

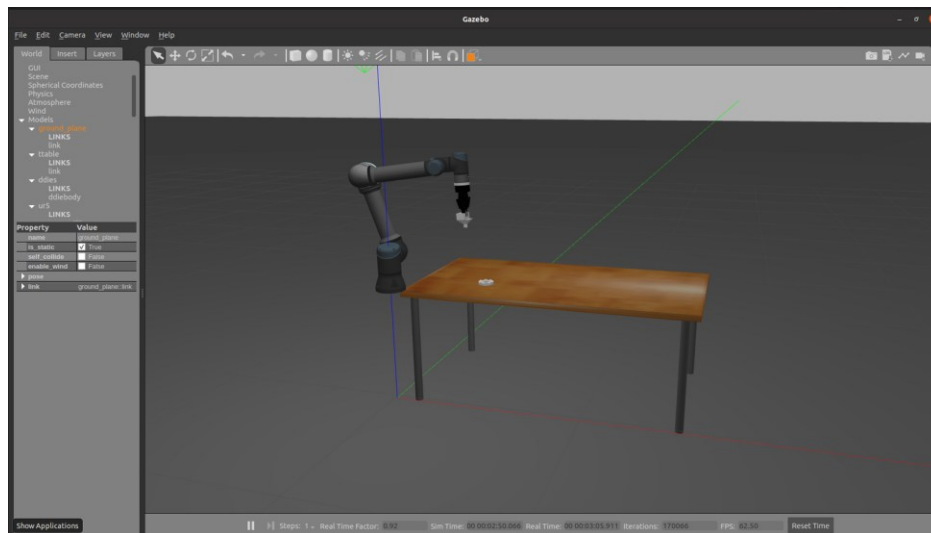


Figure 91 Simulation environment of real scanning scenario with transducer holder and die.

The transducer holder was imported into Gazebo simulation via .stl format (shown in Figure 92). At first, the main part of the cover was imported to urdf file and attached to the gripper finger tip link via a fixed joint. After updating the moveit and srdf file, the holder link can move with the robot.

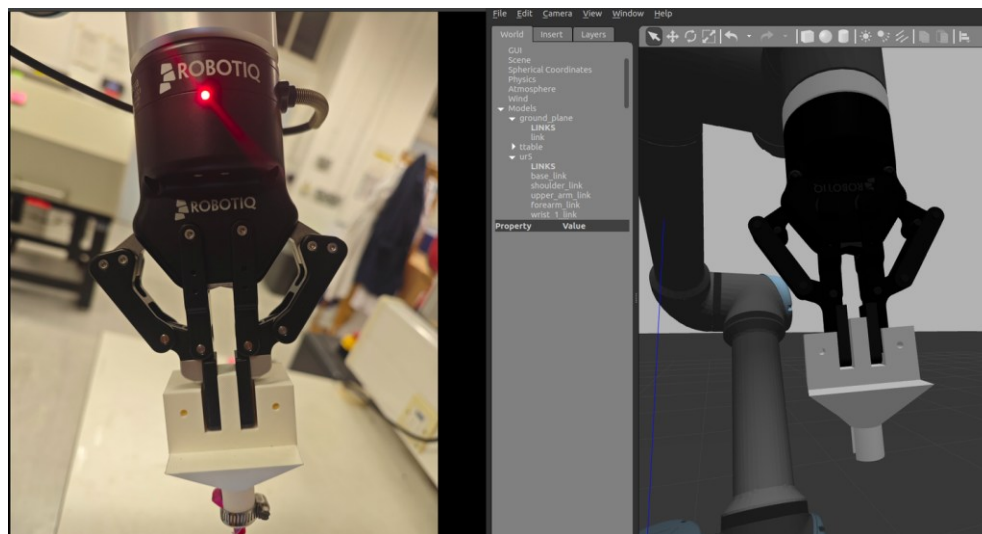


Figure 92 Real world (left) and simulation environment(right) of transducer holder and robot.

After attach the main part of the holder to the gripper, the other parts, i.e., the cover of gripper and the cover of transducer of the holder were fixed on to the gripper (shown in Figure 93). The whole holder set will move with the end-effector of the robot.

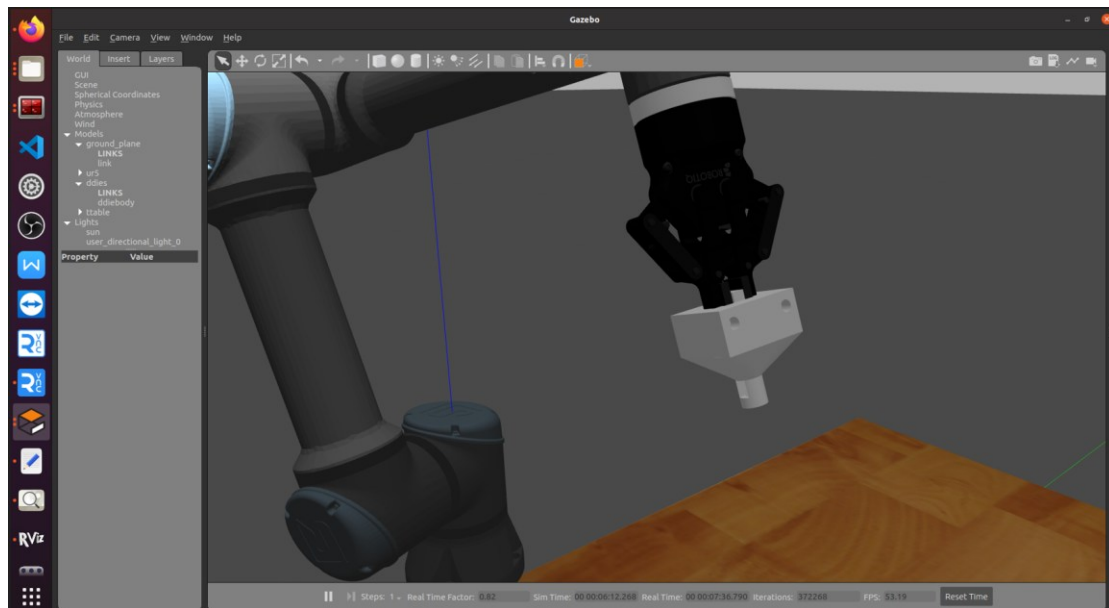


Figure 93 Full set of the transducer holder in Gazebo simulation environment.

For the scanning object, the normal vectors of the die surface can be also calculated using the interpolation algorithm (shown in Figure 94). The calculated normal vectors can be used to calculate the angle difference between the actual end-effector's orientation and the normal direction on the die top surface.

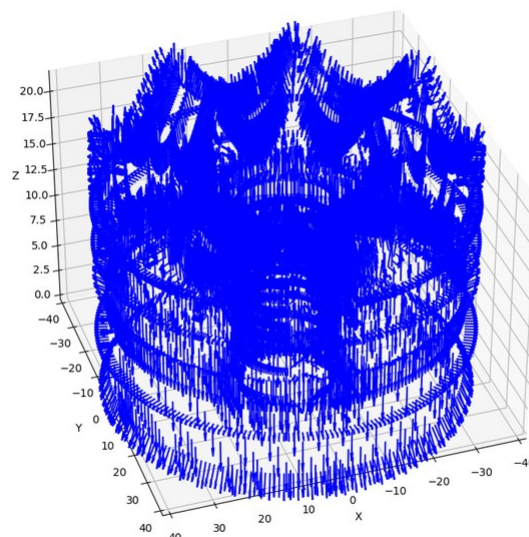


Figure 94 Normal vector of the die surface.

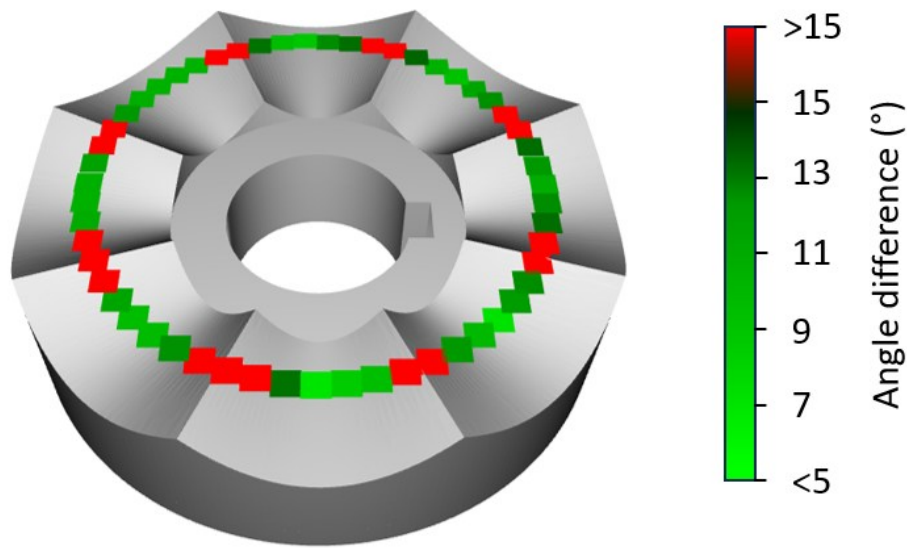


Figure 95 Accuracy of scanning on die surface.

In the simulation task, change the path to circular path along the die. Load the best trained RL model in the curved wing scanning task, and implement the scanning task on the die again. From Figure 95, it can be seen that the angle difference became bigger at the edge of the die. The edges of the die have the biggest peak points of the curvature and the biggest derivative of the normal vector, which will lead to the challenge of control. The average of the angle difference is 12.2299 degrees. The maximum difference is 15.4924, and the minimum is 4.7129. In this study, the edge point is not the research topic, it can be seen from the overall performance, that the end-effector had a bigger angle difference at the edges, but it came back and aligned with the normal vectors of the surface in the next stage scanning task, which resulted in an overall good performance.

6. Experiment

6.1 Equipment and preparation of experiment

Experiment in this study is to validate the results of simulation tests, since the UT part of the task cannot be simulated in the current simulation environment. The experiment should be carried out to validate the robotic control function. The experiment should include the functions such as: real robotic arm compliance control implementation, manual and robotic UT inspection, manual vs. robotic UT comparison, RL robotic UT validation. For experiments in the real world, hardware is needed as follows: UR5e robotic arm, which includes the main body of the 6 DOF robot and the ROBOTIQ 2F-85 gripper. With UR5e robot, a force/torque sensor is embedded in the last link of the robot, the contact force and torque around the end-effector can be measured. The teach pendant that is the original control panel of the robot. The teach pendant is also needed to realise the synchronisation of the external control and the real robot. With the teach pendant, random control of the robotic arm and emergency stop of the robot is possible. The control box of the robot is the assemble of all power electric and control signal hardware. The communication Ethernet cable is connected to the port in this box.

The control computer is with an Intel Core i7 processor. NVidia RTX 3070 GPU is used as the acceleration of the deep learning algorithm. The control computer is connected with the robot via an Ethernet cable. In the teach pendant, a firmware of “External Control” was installed to realise the control from ROS.

Besides robot and control computer, ultrasonic probe and data collection device is needed. Since single-element ultrasonic probe is used in this study, the Olympus EPOCH 650 is used as the data collection device. Since the surface of the object is curved, a delay-line probe, V203-RM, is chosen as the probe. All the hardware needed in this thesis is listed in Figure 96.

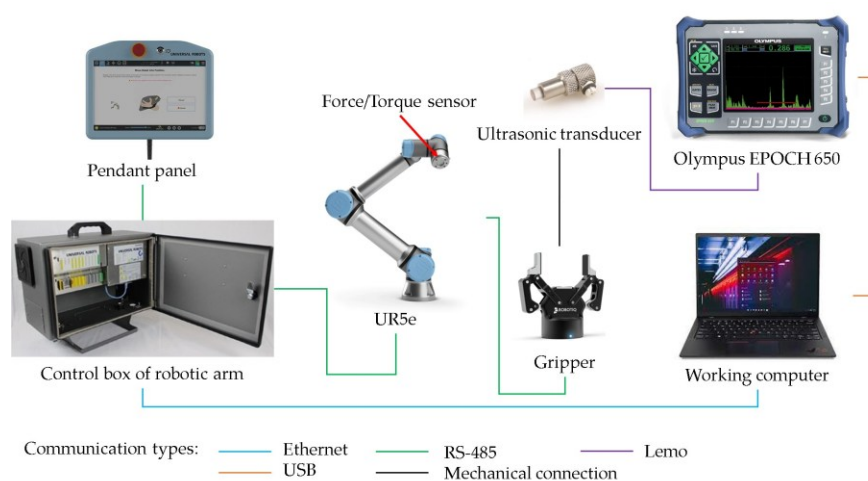


Figure 96 Hardware used in simulation and experiment in this study.

The V203-RM probe is a small-sized single-element delay-line probe, which is suitable for curved surface and detection of subsurface cracks. The centre frequency of V203-RM is 10MHz. The diameter of the element is 0.125 inches (3.175 mm), the diameter of the case is 0.39 inches (9.906 mm), which is the smallest delay-line probe from Olympus. As shown in Figure 97, the probe, the Lemo interfaced wire and couplant was prepared. Olympus EPOCH 650 is the latest data collection device for single-element probes. The EPOCH 650 is suitable for most kinds of single-element UT inspections, including mechanical testing. The LEMO interface is included, and the A-scan amplitude can be recorded via a USB cable to the control computer. The recorded data is in .csv format. Before planning the experiment, the recording function of EPOCH was tested. The probe was also tested on the real object, as shown in Figure 97, the probe can detect the subsurface defects at 8 mm depth.



Figure 97 Ultrasonic transducer V203-RM and data collection equipment EPOCH 650.

Before the purchase of ultrasonic transducer, the comparison of different probes was carried out (see Figure 98). The contact transducer M109-RM, and delay line transducer M201-RM and M203-RM were compared. It can be seen that M203 with 10MHz frequency had the better results, so the delay line probes are more recommended for this task.

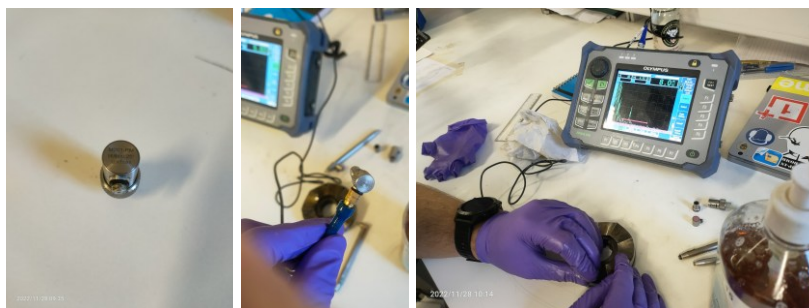


Figure 98 Ultrasonic transducer MR201 and MR203 with equipment EPOCH 650.

A holder is designed to connect the ultrasonic transducer and the gripper of the robot. The purpose of designing the holder is to realise robotic ultrasonic testing with no disruption of the other tasks that the robotic arm engaged with. The holder can be directly installed on the fingers of the gripper, and it can be uninstalled anytime. A spring damper is installed in the holder to prevent harsh contact between the transducer and the target surface (seen in Figure 99). The

picture of the holder installed on the gripper can be shown in Figure 100. The z-axis position of the holder is secured by closing tight of the gripper and the design depth of the slots for the gripper fingers is just the length of the finger. The x,y-axis movement is prevented by fixing the holder on the gripper with screws. The transducer is locked by a cover and the cover is placed at the end of the holder.

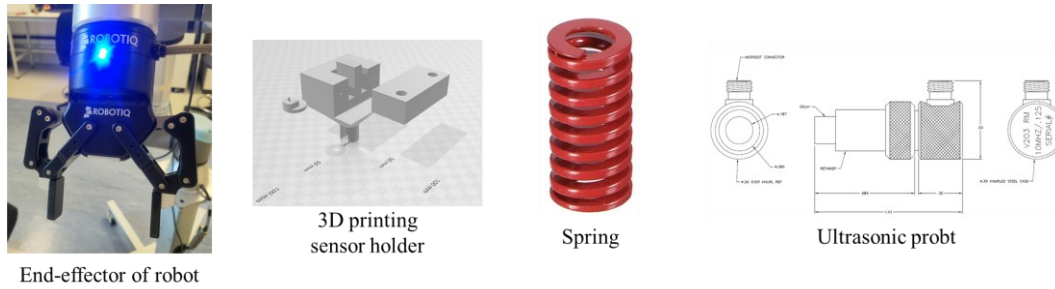


Figure 99 The structure of transducer holder.

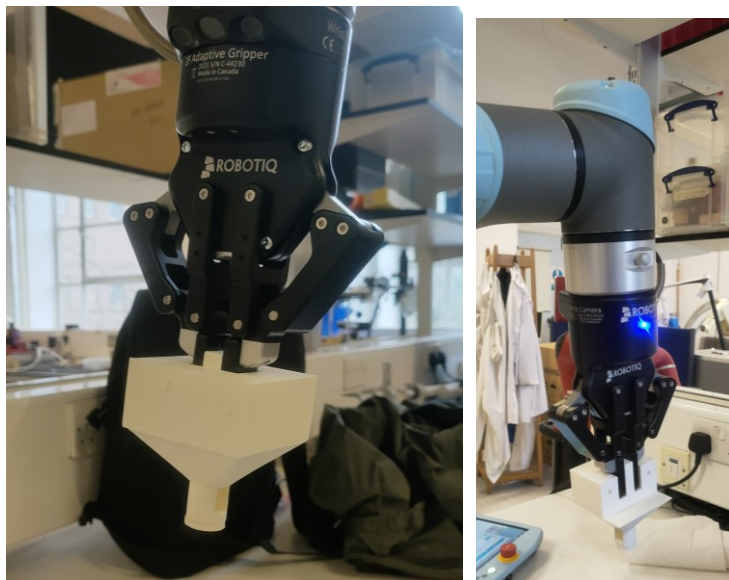


Figure 100 The picture of the 3D printed holder installed on the gripper.

The holder is manufactured by 3D printing from advanced prototyping facility (APF) at Aston University. The reason to use 3D printing is that the holder of the transducer is complex due to the transducer needs damper and the cabling needs special design. If using metallic part, it will be too complicated to design and produce. Considering the weight, the 3D printed part is much lighter than the metallic one, the total weight is 85 g in this 3D printed case. The density of steel is 8 times of PLA, due to design reason, the weight of metallic part can be over 10 times of the weight of PLA. As the load range of UR5e is 5N, using metallic transducer holder is a burden to the robotic arm. More information is introduced in the Appendix A. After manufacturing, the transducer holder can be assembled with the transducer (the assemble seen in Figure 101).

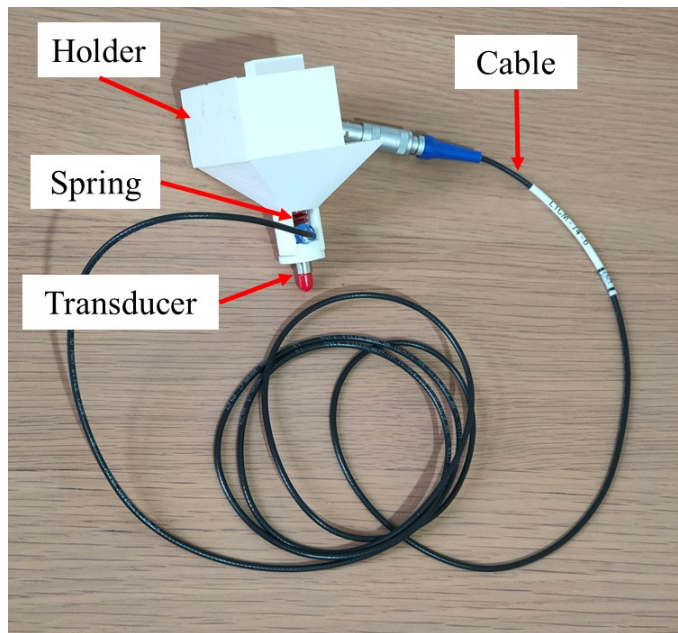


Figure 101 The picture of the transducer holder with the transducer and cable.

The real object in this study is chosen as a real metallic die (as shown in Figure 102). The real object has a small area, but a complex upper surface. It can be used to demonstrate the application of the proposed method. 4 internal artificial defects, i.e., flat bottom drilled hole (FBH), is carried out on the object to imitate subsurface cracks. The internal structure is shown as Figure 102. 4 FBHs, i.e., 2 holes are horizontal group, the other 2 are vertical group. Each hole has the same features, 3mm in diameter, 10mm in depth. For every 2 holes in a group, one hole is underneath the lowest point of the curved surface, the other one is underneath the peak point of the curved surface. The design is to verify the capability of robotic UT. For UT of internal defects, the horizontal defects in this case should be simpler to detect, but the vertical defects are difficult. And the defects under the lower point of curved surface should be easier to detect since the upper surface is close to flat. But the defects underneath the peak point of curved surface are difficult to detect, since the upper surface is more complex, and the trajectory of the probe will be more challenging.

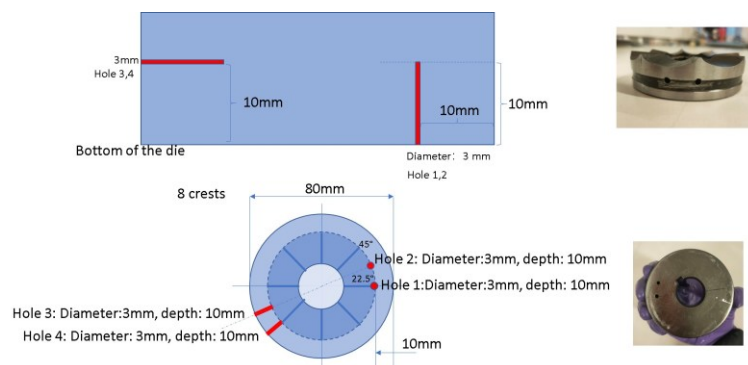


Figure 102 Test object for the experiment.

Experiment 9(E9): contact force measurement on the real robotic arm on the flat surface. Even though it is simpler to detect the defects from the lower flat surface of the die, it is conventional that implementing UT from the upper surface. The reasons include that the upper surface is closer to the defects, and the upper surface is more accessible than the lower surface. Even during the regular maintenance, the upper surface can be accessed using the robot for all kinds of dies/moulds.

As shown in Figure 103, the FT sensor can perceive force from the x, y, and z axes. The gravity components on three axes will be estimated and then help to improve the accuracy of force measurement. And according to the force measurement on each axis, the target orientation can be calculated using the forces at the location (x, y, z).

To implement the orientation optimisation on real arm, model-based orientation optimisation method: Similar to the case in this thesis, researchers from Taiwan developed a method based on RL^{136 136 195}.

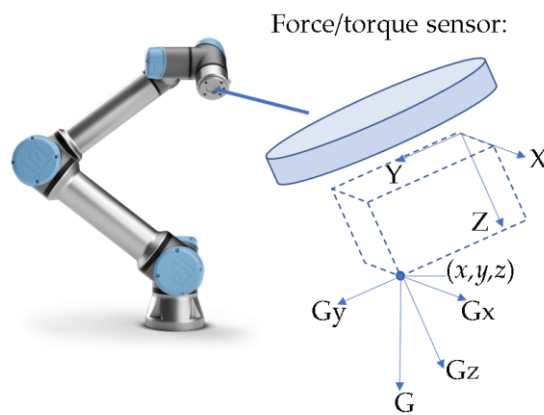


Figure 103 The robotic arm UR5e used in this study and the gravity component at the FTS.

In Figure 104, the real arm contact forces were measured in the approach of the table surface. It can be seen that when the end-effector touched the table surface, the force on the z-axis increased to -30N, while the torque about the z-axis was stable at 0, which means it has no rotation motion around the z-axis. In the meantime, the forces on the x and y-axis had different variance and the torque about the x and y-axis had changes too. It indicates the end-effector was not perpendicular to the touch point and there was rotation motion around the x and y-axis at the 6DOF TF sensor. According to the measured force, the target orientation of the end-effector can be calculated. The adjustment around the y-axis should be F_x/F_z , the adjustment around the x-axis should be F_y/F_z . The actual situation before and after the adjustment can be shown in Figure 105.

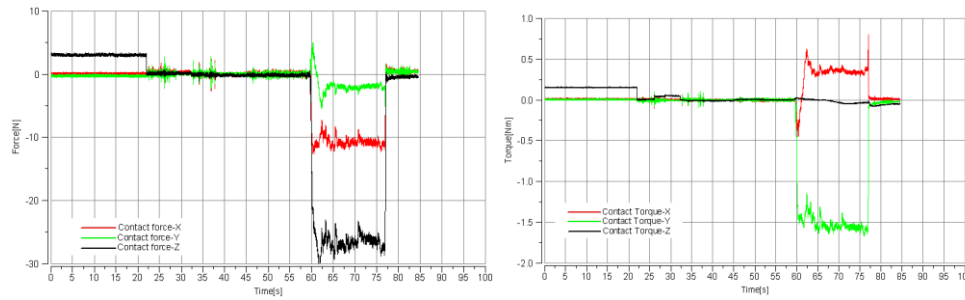


Figure 104 The measurement of contact force and torque of the end-effector of real robot.

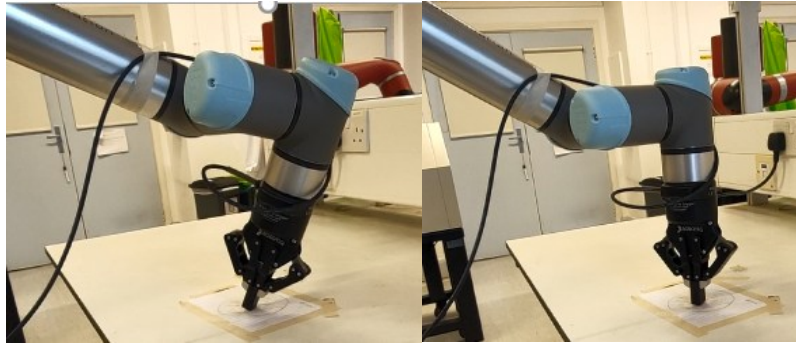


Figure 105 The end-effector before and after the orientation optimisation.

Gravity component affects the force measurement in robotics. Gravity component applies force on the z-axis. If the robotic arm is tilted, gravity also affects the X-axis and Y-axis components. The torque readings will also include gravitational effects due to the mass distribution. The effect of gravity can be shown in Figure 106. The approaches to the table surface was implemented two times, the force was measured. The fuchsia colour curve is the measured force without gravity compensation and was measured in another measurement. When approaching the table surface, the contact force had a big oscillation as it touched the table surface and the controller of robotic arm switched from the force controller to the compliance controller, then the scanning began. During the scanning, the average force was -14.14N without compensation, while the average force was -54.92N with the gravity compensation, which is closer to the target. Therefore, gravity compensation will be applied in this study.

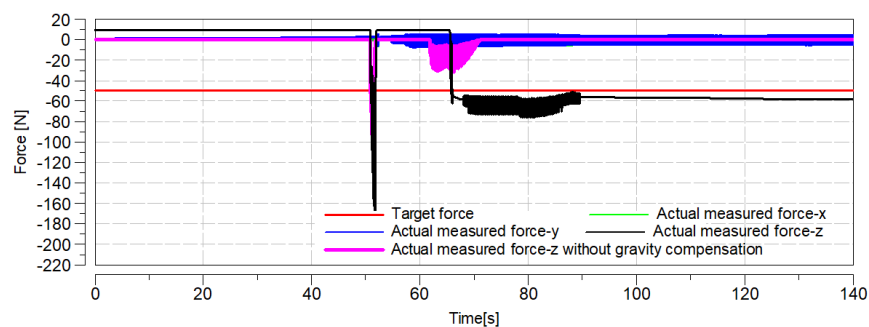


Figure 106 Comparison between with/without gravity compensation.

6.2 Transferring RL controller to real-world

As discussed before, due to safety reasons, RL models are limited to simulation environment in training phase. Due to the features of RL, the agent searches the optimal actions by itself, the random movement of the robot may cause safety issues. Even after the training is finished in simulation, since there is more noise of the signal in observation in the real world, there will be more unexpected objects and scenarios, directly transferring the model to this environment is dangerous for the operators and risky for the hardware. Therefore, transferring RL to real world (sim2real) is a challenging topic, the real world environment is more complex and totally different from the simulation one ¹⁹⁶. The differences between simulation and real world will result in two kinds of differences: different observations and different system dynamics. Methods, such as, extracting features in common between tasks (domain adaptation), enriching training experience (domain randomisation), multitask learning, continual learning and fast learning can be used in the transferring tasks. Other methods, such as, more advanced simulation software, adding real feedback to simulation environment, can also help improve the sim-to-real performance.

Domain adaptation is a method used for the same task in different domains. In prior literature, domain adaption can be divided into two kinds: active and passive domain adaptation. In active domain adaptation, noises were added to the simulated actions or sensors to simulate real world to transfer the simulated model to reality. Simpler policies will perform inefficiently in reality since the environment condition is too complicated ¹⁹⁷. In their article, low-fidelity simulation software SimSpark was used to compare with high-fidelity Gazebo. The results showed that Gazebo was closer to the real world. Gazebo was also used as a surrogate of the real world to implement sim2sim firstly and prepared for the sim2real. In passive domain adaptation, the researchers used simulation software to surrogate the reality as well as possible and transfer the policies without any modifications ¹⁹⁸. The method used in the article can be considered as domain adaption.

For domain randomisation, tools like RoboMaker from Amazon Web Services or the Unity Machine Learning Agents (ML-Agents) toolkit provide capabilities for domain randomization, allowing you to generate diverse training data that is more robust to real-world variations. Simulation has been implemented on mobile robot ¹⁹⁹. The shortcomings of the method were prolonged training time, nondeterministic which means the repeatability of the method is not satisfactory.

Inverse dynamics method should address the discrepancies between source domain and target domain. Forward dynamics model is usually used in this method to predict the actions taken in certain observations. The data from real forward dynamics model should be collected before

training, therefore, the training phase is expensive.

Progressive neural network (PNN) was developed to solve continual and multi-task transferring. PNN is less sensitive to the hyperparameters. Using LSTM makes the method less forgetful about the hyperparameters²⁰⁰. However, the downside of PNN is that the parameters in the algorithm increase exponentially when the number of tasks increases, so that the computational cost will be more.

Transfer learning and meta-learning can transfer RL from simulation to reality. Transfer learning is effective when there is sufficient overlap between the simulation and real-world domains. It can be easier to implement and requires fewer computational resources compared to meta-learning. However, transfer learning may face challenges due to the reality gap, where the differences between simulation and reality can degrade the performance of the transferred policy. There are specialised libraries and frameworks that focus on transfer learning in RL, such as TensorFlow Agents (TF-Agents) and TRFL (Transfer from Lab to Field). These libraries provide tools, techniques, and pre-trained models specifically designed for transferring RL from simulation to reality.

Meta-learning, or "learning to learn," focuses on training models that can quickly adapt to new tasks or environments (see Figure 107). In the context of RL, meta-learning can be used to train an RL agent in a distribution of simulated environments that capture various aspects of the real-world task. The agent learns a meta-policy that can generalize to new tasks or environments more efficiently. During the transfer to reality, the agent adapts its meta-learned policy using a few real-world samples, which allows for faster learning and better performance in the real environment. This method can transfer the prior knowledge from the simulation environment to reality with a few new working samples and prevent overfitting with new data. Meta-learning explicitly focuses on adapting to new environments, which can be beneficial when the real-world domain presents significant variations or challenges not captured in the simulation. Meta-learning typically requires more complex algorithms, extensive training, and a larger number of simulation environments for effective generalization. The learning from simulation to reality phase is fast when the reality task is similar to the simulated one. Currently, there are two kinds of meta-reinforcement learning methods: gradient-based and recurrence-based. After all, the state-of-the-art methods used in transferring sim2real are domain adaption, domain randomisation, inverse dynamics, PNN and Meta-RL.

For the simulation platform, there are cases using ROS+Gazebo to make the simulation¹⁹⁷. Grounded action transformation (GAT) algorithm was designed to leverage the simulated data to learn policies which are useful in reality. Gazebo was also used in the computer vision RL transfer task²⁰¹. The proposed method can not only learn image representations, but also the

control policies based on the image information. The image in the real world was acquired by copying the joint angles in the Gazebo simulation. The image pairs from the simulation and real-world at the same pose were compared to train a model to regress the 3D gripper pose. From a supervised method, the control model was evaluated, and a proposed unsupervised transferring method was also carried out to check the effectiveness of the alignment method.

According to the literature review, the Meta-learning+Domain randomisation is an optimal method to tackle sim2real problem. Meta-learning was used in the transferring task²⁰². In meta-learning, the purpose is to use less previous experience to adapt to new tasks. Meta-learning assumes that the new tasks are similar to the previous tasks. Meta-learning was developed to solve the problem that machine learning algorithms are not suitable for all domains. Some algorithms are suitable for one domain, but cannot perform well on another domain. A domain in machine learning is including all the parameters and data that can make a context, or a function.

$$D = P\{X, P(X)\} \quad (67)$$

where D is the domain, X is the inputs, and $P(X)$ is the probability distribution. Meta-learning is designed to tackle this problem. For transferring sim2real, the problem can be considered as a problem transferring between different domains. However, meta-reinforcement learning has a disadvantage that it does not consider observation discrepancies between simulation and reality.

Currently, most of prior literature mentioned about transferring from simulation to reality, but less of them published their codes, which adds a bit of difficulties to replicate their algorithms. Some tools that can be used for sim2real are open-sourced. Habitat, for example, is a high-level library for the tasks like transferring²⁰³. Gibson's environment, on the other hand, focuses on the rendering of the environment to get closer to reality²⁰⁴.

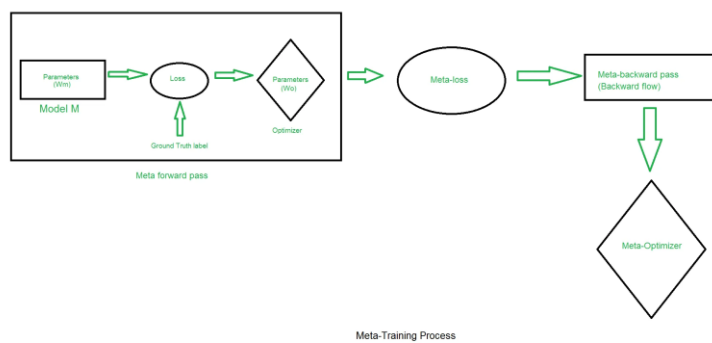


Figure 107 Illustration of meta-learning.

Due to time limitations, only domain adaption transfer is carried out in this study. More adaptive transfer will be studied in future research. To implement the transfer from simulation to the real-world, the classic transferring method, “pre-training followed by Fine-Tuning”, is used. The pre-trained RL model in a simulation environment can be saved and then fine-tuned with real-world data to adapt to the target domain. The current solution is to transfer the simulated solution to the real world. Since the real-world robot location is not the same as the simulation.

6.3 Implementation of robotic inspection

Since the RL training of simulation model has been finished, the transfer from simulation to real-world is finished, all the hardware and software for the robotic inspection have been done. The implementation of the robotic inspection can be started. In this sub-section, the real robotic inspection using ultrasonic transducer will be implemented.

To get a pre-trained model, the best reward-scored model should have been saved. Then, in `stable_baselines3` code, load the model in the modified real-world environment. In this case, the `best_model` in simulation environment should be loaded into the real-world environment.

To log the data in EPOCH650, a series of software named “GageView” will be used. In Epoch650, live videos can be exported to external storage. After the export, the video is in the format of “.vid”, which is in a special code that the video players cannot play. A transformation software “GageView Video” should be used to convert the .vid to .avi format. The video can be presented in the presentation.

The EPOCH650 was set up as shown in Figure 108, then the device was calibrated firstly (see Figure 109). The reflective data can be exported to PC via USB communication, while the MicroSD card can be used to transfer screenshots and videos. GageView Pro is needed to save the raw data (see Figure 110), and using the code “`param_RawData=0,x`” (x is the range of detection), the raw data will be exported. With the setups, the raw data can be logged as shown in Figure 111.

The “x” in the code indicates the maximum limit of the range of the detection. It should be equal to or smaller than the range set on EPOCH device. The difference between $x=10$ and $x=30$ at the same place can be seen in Figure 111.

Experiment 10(E10): this experiment is the final part of the study to test the proposed robotic UT on the real object. To verify the working process or robotic UT, the UT on flat surface was firstly implemented on the die (shown in Figure 112). It uses compliance controller with

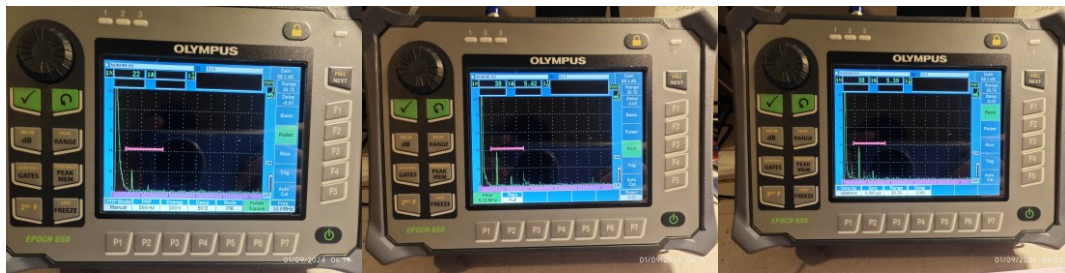


Figure 108 Settings of EPOCH 650.



Figure 109 Calibration of the transducer.

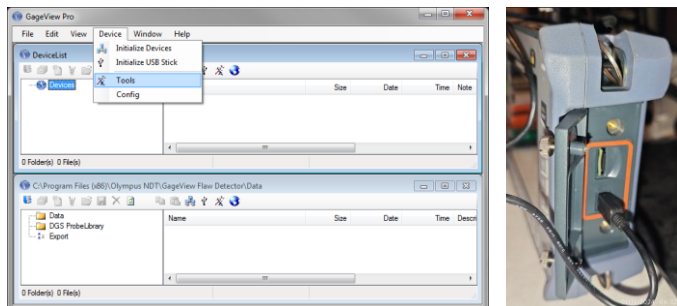


Figure 110 Screenshot of the software GageView Pro (left). USB and MicroSD card interface on Epoch650 (right).

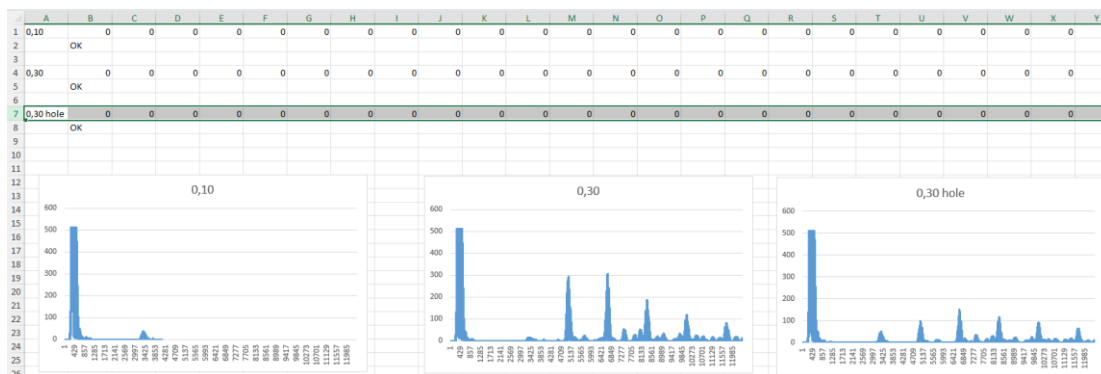


Figure 111 Screenshot of the data and curve of the saved data from EPOCH650.

target_wrench -50N, and the path of the end-effector is a circle with a radius of 5cm. At each position, the thickness was measured with EPOCH650, the results are shown in Figure 113.

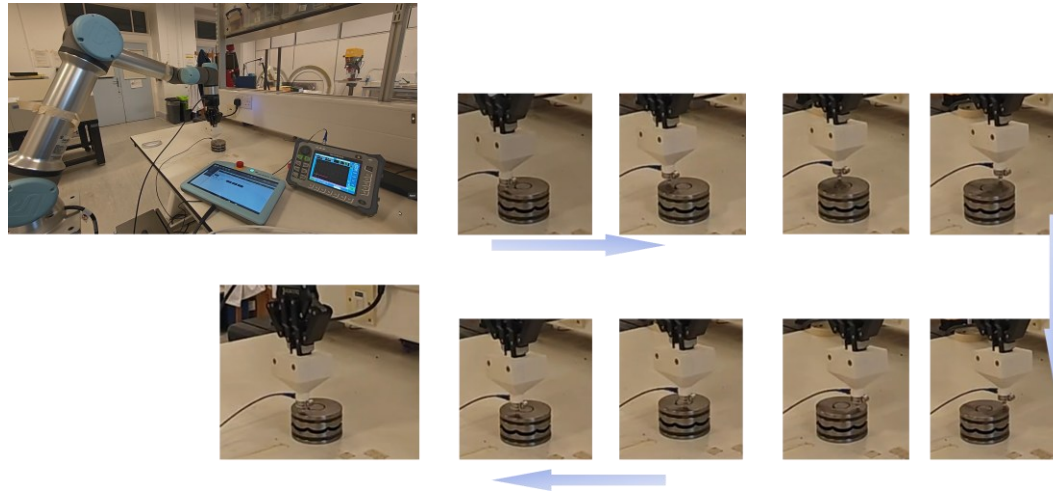


Figure 112 Screenshot of the robotic UT on flat surface.

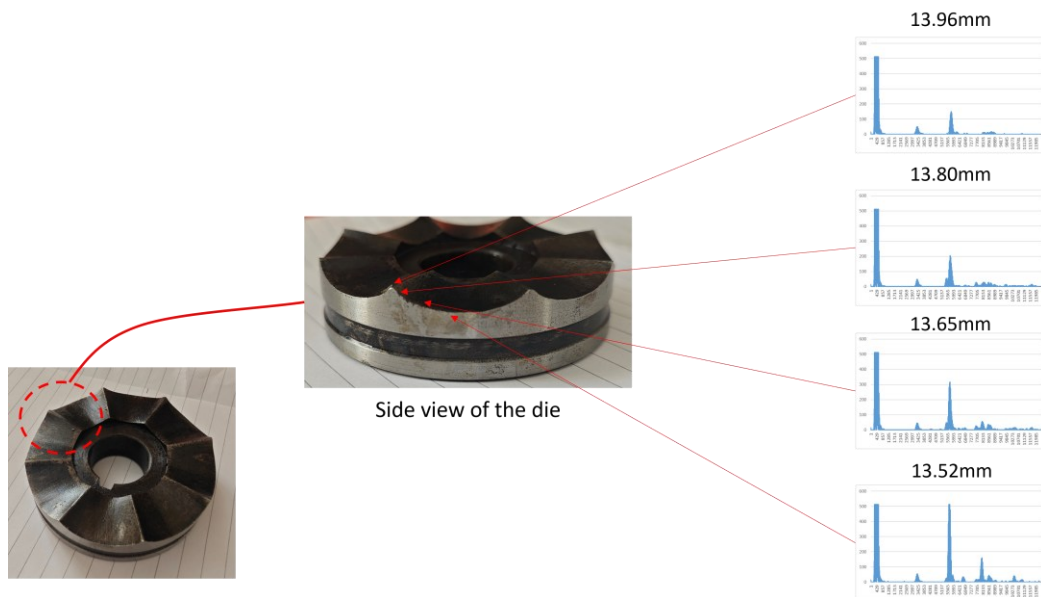


Figure 113 UT raw data of thickness measurement around the surface of the die.

After the trial on the flat surface, the experiment on the complex surface is implemented. As all equipment and software have been set up, a complete experiment scenario can be presented as shown in Figure 114. In this figure, it is shown that the whole set of hardware: UR5e robotic arm gripping the holder with the ultrasonic transducer contacting the target die. The robot is communicating with ROS and the transducer is displayed in Epoch650. Couplant is also implemented on the dies, as the bottle can be seen in the figure.



Figure 114 Photo of experiment scenario with Epoch, PC, Robotic arm and transducer.

During the experiment stage, it was found that the inspection of ultrasonic probe had much noise. After check of the probe and the cable, it was found the cable had a grounding issue, which lead to much noise even when the probe was not connected. Due to transducer problem, the probe and cable were sent back to Olympus Europe to repair. During this period, a pencil was fixed at the transducer holder to continue the experiment (as shown in Figure 115). The target contact force is -20N, and the average of the measured force is -20.4494N, with the error

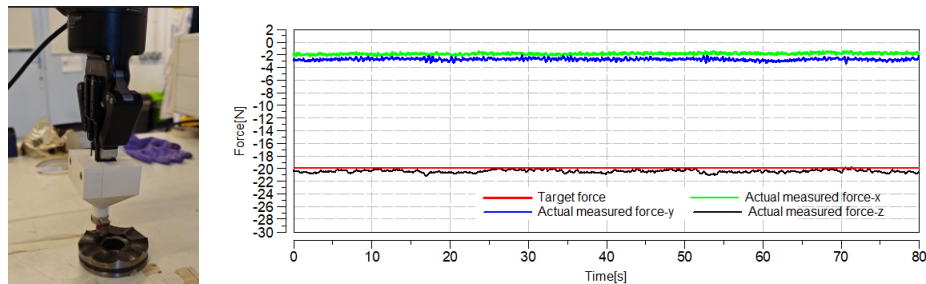


Figure 115 Screenshot of scanning experiment.

of $\pm 2.24\%$ which is within the limit of expectation (2.5%). The standard deviation is 0.2327, which is better than the simulation situation, and acceptable in the real-world experiment.

The processes for the transducer scanning the complex surface are shown in Figure 116. It is the screenshot of end-effector scanning over each waypoint on the die top surface. The target contact force is also -20N. The video of scanning can be checked on the Box share of the university.

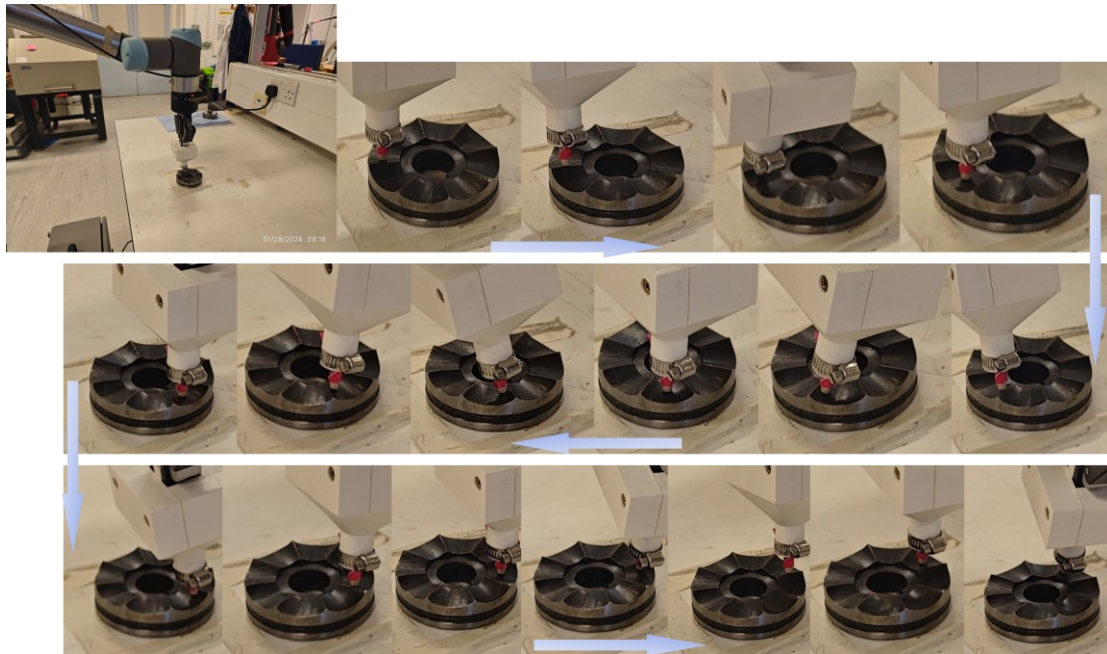


Figure 116 Scanning of complicated surface of the die.

6.4 Results

The results of proposed inspection method are compared with manual inspection and conventional robotic inspection on the same die are shown in Table 11. Since the experiment in this study is to validate the optimised control of robot and the RL training, it was not focused on the detection of size of the defects, but only the detection of the defects.

To validate the optimisation of control in simulation, the experiment was carried out. The comparison between manual UT inspection, conventional compliance controller and RL+ compliance controller methods are implemented. The efficiency and accuracy were mainly checked. For efficiency, the total time used during inspection was logged. For each method, several times of inspection were carried out and the efficiency values are the average value. For accuracy, since the UT probe used in this study is only a single-element conventional probe and the probability of detection was not much studied in this study, the value of accuracy mainly takes the detection of defects in consideration. Because the manual inspection was limited to the method, that it cannot provide accurate position information of the defects, the accuracy

value cannot be valued.

It should be noticed that the length of efficiency time is only for scanning, but not including training time for the robotic scanning. And the accuracy of manual inspection cannot be assessed since manual inspection has too much flexibility and cannot log the coordinates of each scanning position.

Table 11 The comparison of inspection results.

	Efficiency	Accuracy
Manual inspection	8 hours	N.A. (cannot evaluate, since manual inspection does not include any position information)
Conventional robotic inspection	4 hours	50%
The proposed method	4 hours	80%

7. Discussion and Conclusion

7.1 Discussion

In this study, an automated inspection method on dies with a robotic arm is proposed. The simulation and control model of the robotic arm were first implemented. This was not only for the thesis but also for all future researchers in the Aston University robot community. The study's contributions include but are not limited to an improved compliance controller on a UR5e robot, which can be used both in simulation and real-world, an RL-trained compliance controller on a ROS platform, and a simulation and real-world RL-guided surface scanning robotic task.

Even though this study's research concerns robotic inspection, the main topic is the robotic control. The proposed robot control method can be used in many other applications, such as robotic polishing, robotic machining, mobile robotic machining, 3D printing, etc. The difference between UT inspection and other applications is that UT will not remove or apply extra materials on the surface. Therefore, for the applications of the proposed method in other industries, more studies on dynamic process control are needed. Some say this study's innovation is not convincing, but I reckon if the research can be used in applications and make value, it is good research.

In addition to building a control model for UR5e on surface scanning, simulation and experiments were also carried out in the study. It can be seen that the RL-guided compliance controller has 40% better force control and 30% better orientation control than the pure compliance controller. Although the evaluation is mainly on the orientation performance, the speed control of the RL controller also affects the orientation performance on the curved surface. The influence of this can be studied in future studies. And the robotic implementation saves more time than manual UT regardless of the training time. In this study, the robotic tasks are only limited to surface scanning operations, if the training research in future can include more tasks or transfer it to other applications, then it will be more valuable even with the long training time. Since only a force/torque sensor was used to keep the simplicity, the tasks can only be small surfaces scanning on whose location is known.

Someone argues that the accuracy of localisation of the defects does not matter. However, as I disclaimed, the localisation of the defects is important as it will decide for future planning of remanufacturing, for example, where to apply SM, how much material should be removed, what feed rate should be for 3D printing. And, the prediction of the defects is also important for predictive maintenance and predictive remanufacturing. If the subsurface defects develop into surface cracks and being fatal, then the repairing will be costly, and the catastrophe brought

to the manufacturing will be massive.

7.1.1 Limitations

The UT inspection was assumed to be perfect in the robotic control. The probability of detection (POD) was not discussed in this study. So, all the control of the robotic arm is implemented for only one time. However, in the real world, sometimes the results of UT are not reliable, and a re-test is needed. The potential future research trend will be including the UT simulation in the closed-loop of robotic UT.

The study is limited to the physics settings in Gazebo simulation, which set the surface of the object and all the objects in the world to rigid blocks, which may sometimes not be the actual case. Research is also trending toward using dynamic soft materials for ultrasonic testing, but since Gazebo's performance is limited, other simulation software (better with UT simulation) can be studied for future research on UT on soft objects.

3D reconstruction using the proposed control method can be studied in the future, and more advanced equipment, such as 3D scanner can be used to reconstruct the object to validate the proposed method. It can be also used to validate the accuracy of locations of the end-effector. The 3D reconstructed model can again, improve the training quality of the RL algorithm.

The objective of this study is to carry out automated inspection without adding extra sensors to the robotic arm. Therefore, only a force/torque sensor is used to comply with the surface. However, this limits the working capacity of the method, it cannot locate the objects or estimate the size of the object. It cannot do path planning, navigating to the objects. This can be improved in the future work.

Due to the limitation of single-element UT, the UT results on edges are not reliable enough. And the actual control on the edges of the object is not ideal. To implement better control and prepare for future research, more sophisticated control on the edges of the complicated objects can be carried out, which will be the next step of the researcher.

RL simulation in this study took days to finish. As a simulation, I was watching the performance, which contains no redundant actions from the robotic arm. However, it can be more efficient when carrying out the simulation experiments, for example, using different time scales in Gazebo.

And for the adaptability of the RL model, it was only trained on a flat surface, curved surface and the actual surface of the object. It only transfers the RL-trained model in flat and curved surfaces into the actual scanning task, which may not be generic enough. However, training on too many scenarios will lead to different problems, such as catastrophic forgetting. Moreover,

training on a more challenging surface will increase the difficulty of finishing the training, which will result in no data. Therefore, it is more important to focus on the ability of the generalisation of the RL model. Transfer learning, on the other hand, can be another potential research in the future, it can be used to transfer the pre-trained model to implement the controller on different tasks, such as different surfaces, and different application scenarios, such as polishing or disassembly, which will be challenging, especially to real industrial application scenarios. Future research from the group is anticipated.

7.1.2 Future work

The compliance controller of this study is good enough to maintain constant contact force; however, the position accuracy can be improved. For further research, obstacle and collision avoidance can be considered since safety is the most critical factor in manufacturing, and the objects for the proposed method can be valuable. A camera can be attached to the robot to navigate to the object, subsequently, implement the compliance surface scanning.

Studies can be continued to consider overall energy consumption and coverage of the object's surface for scanning path planning. I have started some research using Postman problem optimisation path planning to plan the path economically; the interesting research was stopped while some researchers considered it unevaluable. However, as the object's surface becomes larger and more complex, path planning can be very valuable. Moreover, the automatic approach from the home position to the object can also be studied to implement fully automatic robotic UT.

In this study, the RL was trained in simulation environments. The next step is to implement RL directly in the real world, i.e., directly train the robot in the real world to adapt more types of objects. Moreover, a potential research direction is to combine supervised learning and RL to improve training efficiency even more. Transfer learning can be used to improve the controller's intelligence with less learning data. The overall purpose is to make the controller solving the robotic scanning task more human-like.

For robotic industrial UT, more complex scenarios should be studied. Different defects on different objects and different depths should be studied to study the POD of the proposed method further. To compare the results, PAUT can be used to improve the efficiency and effectiveness of the method. If possible, it can be a very interesting study to combine the robotic simulation platform (ROS) with the UT simulation platform, such as CIVA. In that case, the robotic UT can be simulated thoroughly and implemented in the real world. As mentioned, the proposed method, RL-guided compliance controller, can be applied to other industrial applications, such as grinding, 3D printing, etc. Moreover, applying this robotic arm onto a

mobile platform and implementing autonomous vehicle control can be applicable to large-size industrial objects, such as turbines.

7.2 Conclusion

In this thesis, an RL-based automated robotic ultrasonic testing method using only force sensing is proposed. At first, a control platform for the UR5e robotic arm, which includes the simulation environment and the implementation in the real world, was established. With the platform, the ultrasonic testing surface scanning task was simulated and carried out in the real world. UT transducer was fixed on the end-effector of the robotic arm with a 3D-printed holder.

Based on the robotic control platform, an improved compliance controller is tuned to maintain constant contact force. The tolerance is within 5%. The controller is used as subsystem of the proposed method. To make the UT task more reliable and efficient, the RL algorithm was used in the control of the robotic arm. With the RL algorithm, a CAD model is not needed for the surface scanning tasks. Only the contact force between the probe and the object's surface is measured to optimise the probe's orientation.

A Python interface between ROS and Stable-Baselines3 has been established. The interface makes the RL training in the ROS simulation environment feasible. It perceived the environment in ROS, trained the model using the Stable-Baselines3 library, and then sent actions back to the ROS simulation environment to close the loop.

The RL trains the robot model for the assigned tasks and the trained model can be carried out in the real world. The RL was simulated in Gazebo firstly, then transferred to the real world. The implementation in the real world proves that the robotic automatic UT has better efficiency and better adaptability than human manual UT and conventional control method. The method gets rid of the constraint of CAD model of the object and can be implemented on any object. The platform established in this study can be used in inspection in manufacturing industry, it can help inspection in remanufacturing and help improve economic benefits for the entrepreneurs. Traditional entrepreneurs may not be interested in this topic since it is a bit far away from realistic manufacturing. However, it can also be used in future research as a basement to develop future robotic contact tasks, such as robotic 3D printing remanufacturing, polishing, etc.

The proposed method has been verified in simulation and experiments. The proposed controller can not only optimise the orientation of the end-effector but also optimise the moving speed to optimise the trajectory of scanning. Compared to conventional compliance control, the overall probe orientation error of the proposed method is much better improved. The error between the actual orientation and the normal direction of the object surface is within 15 degrees.

Future research topics could include integrating a mobile platform into this robot, a mobile robotic inspection platform can be carried out for a more significant object and a flexible application scenario. Since UT inspections are typically implemented on high-added-value objects, such as high-end dies and moulds and aeroplane components, sometimes the size matters. If the surface contact implementation can be extended to the full size of a large object, the value of this research will be enlarged, moreover, with mobile platforms, more research topics can be involved.

ROS2 can be tried out for more real-time control. More unknown objects and surfaces can be explored, and more meta-learning algorithms can be developed to transfer from simple simulation scenarios to complicated reality applications.

If possible, UT simulation software, such as CIVA, can be implemented via an API in the platform to finish the close loop of the UT simulation. At that point, the RL training can be evaluated by the real-time UT inspection results, which will be reliable and closer to the real world.

Reference

1. Lu, Y.; Zheng, H.; Chand, S.; Xia, W.; Liu, Z.; Xu, X.; Wang, L.; Qin, Z.; Bao, J., Outlook on human-centric manufacturing towards Industry 5.0. *Journal of Manufacturing Systems* **2022**, *62*, 612-627.
2. Our World in Data GDP per capita, 1820 to 2018. <https://ourworldindata.org/grapher/gdp-per-capita-maddison-2020>.
3. Kennedy, P., *The rise and fall of the great powers: economic change and military conflict from 1500 to 2000*. Vintage: 2010.
4. Kelly, M.; Mokyr, J.; Ó Gráda, C., The mechanics of the Industrial Revolution. *Journal of Political Economy* **2023**, *131* (1), 59-94.
5. Jones, P. M., Industrial Enlightenment: Science, technology and culture in Birmingham and the West Midlands, 1760–1820. In *Industrial Enlightenment*, Manchester University Press: 2017.
6. Allen, G. C., *The industrial development of Birmingham and the Black Country, 1860-1927*. Routledge: 2018.
7. Terry-Chandler, F., Compulsory industriousness: working conditions and exploitation in Birmingham during the Industrial Revolution. *Midland History* **2019**, *44* (1), 71-84.
8. Christophers, B., The rentierization of the United Kingdom economy. *Environment and Planning A: Economy and Space* **2023**, *55* (6), 1438-1470.
9. Statistics, O. f. N., EMP13: Employment by industry. <https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/employmentandemployeetypes/datasets/employmentbyindustryemp13>, 2023.
10. Foresight, U., The future of manufacturing: a new era of opportunity and challenge for the UK. *Summary Report, The Government Office for Science, London* **2013**, *20*.
11. Graedel, T. E.; Erdmann, L., Will metal scarcity impede routine industrial use? *MRS bulletin* **2012**, *37* (4), 325-331.
12. Bloodworth, A.; Gunn, G., The future of the global minerals and metals sector: Issues and challenges out to 2050. *Geosciences: BRGM's journal for a sustainable Earth* **2012**, *15*, 90-97.
13. Vieira, L. C.; Longo, M.; Mura, M., Are the European manufacturing and energy sectors on track for achieving net-zero emissions in 2050? An empirical analysis. *Energy Policy* **2021**, *156*, 112464.
14. Ramakrishna, S.; Ngowi, A.; Jager, H. D.; Awuzie, B. O., Emerging industrial revolution: Symbiosis of industry 4.0 and circular economy: The role of universities. *Science, Technology and Society* **2020**, *25* (3), 505-525.
15. King, A. M.; Burgess, S. C.; Ijomah, W.; McMahon, C. A., Reducing waste: repair, recondition, remanufacture or recycle? *Sustainable development* **2006**, *14* (4), 257-267.
16. Philipp, S.; Fischer, S.; Drews, M.; Ton, B.; Jürgen, K.; Linda, K.; Karoline, A., Evidence-based narratives in European research programming. *European Journal of Futures Research* **2021**, *9* (1).
17. Graham, I.; Goodall, P.; Peng, Y.; Palmer, C.; West, A.; Conway, P.; Mascolo, J. E.; Dettmer, F. U., Performance measurement and KPIs for remanufacturing. *Journal of Remanufacturing* **2015**, *5*, 1-17.
18. Rejeb, H. B.; Zwolinski, P., Development of educational contents on circular economy and critical raw materials challenges. *Procedia CIRP* **2020**, *90*, 759-765.

19. Zacharaki, A.; Vafeiadis, T.; Kolokas, N.; Vaxevari, A.; Xu, Y.; Peschl, M.; Ioannidis, D.; Tzovaras, D., RECLAIM: Toward a new era of refurbishment and remanufacturing of industrial equipment. *Frontiers in Artificial Intelligence* **2021**, *3*, 570562.
20. Bennett, J.; Garcia, D.; Kendrick, M.; Hartman, T.; Hyatt, G.; Ehmann, K.; You, F.; Cao, J., Repairing automotive dies with directed energy deposition: industrial application and life cycle analysis. *Journal of Manufacturing Science and Engineering* **2019**, *141* (2).
21. Veligorskyi, O.; Chakirov, R.; Khomenko, M.; Vagapov, Y. In *Artificial neural network motor control for full-electric injection moulding machine*, 2019 IEEE International Conference on Industrial Technology (ICIT), IEEE: 2019; pp 60-65.
22. Yeo, N.; Pepin, H.; Yang, S., Revolutionizing technology adoption for the remanufacturing industry. *Procedia CIRP* **2017**, *61*, 17-21.
23. Saboori, A.; Aversa, A.; Marchese, G.; Biamino, S.; Lombardi, M.; Fino, P., Application of directed energy deposition-based additive manufacturing in repair. *Applied Sciences* **2019**, *9* (16), 3316.
24. Thompson, S., *Handbook of mould, tool and die repair welding*. Elsevier: Cambridge, England, 1999.
25. Chen, C.; Wang, Y.; Ou, H.; He, Y.; Tang, X., A review on remanufacture of dies and moulds. *Journal of Cleaner Production* **2014**, *64*, 13-23.
26. Wollenhaupt, G., Iot slashed downtime with predictive maintenance. *PTC*, [Электронный ресурс]. Доступно: <http://www.ptc.com/product-lifecycle-report/iot-slashes-downtime-with-predictive-maintenance>, accessed March **2017**, 7.
27. WILLIAMANDREW PUBLISHI, N., Handbook of mold, tool and die repair welding.
28. Leptidis, S.; Papageorgiou, D.; Medrea, C.; Chicinaş, I., Failure analysis of an EDM machined mold-printing die used for the production of truck spare parts. *Engineering Failure Analysis* **2016**, *61*, 62-68.
29. FDA FDA alerts consumers of nationwide voluntary recall of EpiPen and EpiPen Jr. <https://www.fda.gov/news-events/press-announcements/fda-alerts-consumers-nationwide-voluntary-recall-epipen-and-epipen-jr>.
30. NHTSA Safety Recall COM – EXPANSION Notice Multiple Models and Model Years Power Window Master Switch (PWMS). <https://static.nhtsa.gov/odi/rcl/2012/RCMN-12V491-7300.pdf>.
31. Stewart, J. The Worst Thing About Tesla's 123,000-Car Recall? Crummy Timing. <https://www.wired.com/story/tesla-model-s-steering-bolt-recall/>.
32. Plasticfailure.com Automotive parts causes deaths. <https://www.plasticfailure.com/case-studies>.
33. Wood, D. A. FORD SHIFTER CABLE BUSHING RECALLS FAILED, ALLEGES LAWSUIT. <https://www.carcomplaints.com/news/2023/ford-shifter-cable-bushing-recall-lawsuit.shtml>.
34. Boots Customer Service Product Recall. <https://www.boots.com/floating-editorial/customer-services/product-recall>.
35. Fraunhofer MEMS hands on access fab. https://www.fraunhofer.jp/content/dam/japan/ja/documents/Events/2019/Presentations/11_Kentaro_Totsu.PDF.
36. Margetan, F.; Umbach, J.; Roberts, R.; Friedl, J.; Degtyar, A.; Keller, M.; Hassan, W.; Brasche, L.; Klassen, A.; Wasan, H., Inspection developments for titanium forgings. *Air Traffic Organization Operations Planning Office of Aviation Research and Development Washington*,

DC 2007, 20591.

37. Wang, Z.; Xu, Y.; Ma, X.; Thomson, G. In *Towards smart remanufacturing and maintenance of machinery-review of automated inspection, condition monitoring and production optimisation*, 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE: 2020; pp 1731-1738.
38. Liu, B.; Liang, Z.; Parlikad, A. K.; Xie, M.; Kuo, W., Condition-based maintenance for systems with aging and cumulative damage based on proportional hazards model. *Value Based and Intelligent Asset Management: Mastering the Asset Management Transformation in Industrial Plants and Infrastructures* **2020**, 211-231.
39. Milan, J. C.; Machado, A. R.; Tomaz, Í. V.; da Silva, L. R.; Barbosa, C. A.; Mia, M.; Pimenov, D. Y., Effects of calcium-treatment of a plastic injection mold steel on the tool wear and power consumption in slot milling. *journal of materials research and technology* **2021**, 13, 1103-1114.
40. Kek, T.; Kusic, D.; Janez, G., Crack detection in injection molds. *Application of Contemporary Non-Destructive Testing in Engineering* **2017**, 1000 (6), 247-253.
41. KUSIĆ, D.; KEK, T.; SLABE, J. M.; SVEČKO, R.; GRUM, J., Use of AE Method for Detecting Quality Of Injection-Mold Engraving Inserts.
42. KAMBER, Ö. Ş.; TAŞÇIOĞLU, E., The Effect of Processing on the Surface and Subsurface Characteristic of Plastic Injection Mold Steel. *International Journal of Advances in Engineering and Pure Sciences* **31**, 17-22.
43. Moreira, E. E.; Alves, F. S.; Martins, M.; Ribeiro, G.; Pina, A.; Aguiam, D. E.; Sotgiu, E.; Fernandes, E. P.; Gaspar, J. In *Industry 4.0: Real-time monitoring of an injection molding tool for smart predictive maintenance*, 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE: 2020; pp 1209-1212.
44. Weinert, A.; Tormey, D.; O'Hara, C.; McAfee, M., Condition Monitoring of Additively Manufactured Injection Mould Tooling: A Review of Demands, Opportunities and Potential Strategies. *Sensors* **2023**, 23 (4), 2313.
45. Georgeson, G. E.; Kollgaard, J. R.; Holmes, T. M., Forming tools and flexible ultrasonic transducer arrays. Google Patents: 2020.
46. Ingarao, G., Manufacturing strategies for efficiency in energy and resources use: The role of metal shaping processes. *Journal of Cleaner Production* **2017**, 142, 2872-2886.
47. Whlean, C.; Sheahan, C., Using additive manufacturing to produce injection moulds suitable for short series production. *Procedia Manufacturing* **2019**, 38, 60-68.
48. Frohn-Sörensen, P.; Geueke, M.; Tuli, T. B.; Kuhnhen, C.; Manns, M.; Engel, B., 3D printed prototyping tools for flexible sheet metal drawing. *The International Journal of Advanced Manufacturing Technology* **2021**, 115 (7-8), 2623-2637.
49. Morrow, W.; Qi, H.; Kim, I.; Mazumder, J.; Skerlos, S., Environmental aspects of laser-based and conventional tool and die manufacturing. *Journal of Cleaner Production* **2007**, 15 (10), 932-943.
50. Karvonen, I.; Jansson, K.; Tonteri, H.; Vatanen, S.; Uoti, M., Enhancing remanufacturing—studying networks and sustainability to support Finnish industry. *Journal of Remanufacturing* **2015**, 5, 1-16.
51. Siddiqi, M. U.; Ijomah, W. L.; Dobie, G. I.; Hafeez, M.; Gareth Pierce, S.; Ion, W.; Mineo, C.; MacLeod, C. N., Low cost three-dimensional virtual model construction for remanufacturing industry. *Journal of Remanufacturing* **2019**, 9 (2), 129-139.
52. Lee, C.-M.; Woo, W.-S.; Roh, Y.-H., Remanufacturing: Trends and issues. *International Journal of Precision Engineering and Manufacturing-Green Technology* **2017**, 4 (1), 113-125.

53. Lee, H. B.; Cho, N. W.; Hong, Y. S., A hierarchical end-of-life decision model for determining the economic levels of remanufacturing and disassembly under environmental regulations. *Journal of Cleaner production* **2010**, *18* (13), 1276-1283.
54. Chau, M. Q.; Nguyen, X. P.; Huynh, T. T.; Chu, V. D.; Le, T. H.; Nguyen, T. P.; Nguyen, D. T., Prospects of application of IoT-based advanced technologies in remanufacturing process towards sustainable development and energy-efficient use. *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects* **2021**, 1-25.
55. Fofou, R. F.; Jiang, Z.; Wang, Y., A review on the lifecycle strategies enhancing remanufacturing. *Applied Sciences* **2021**, *11* (13), 5937.
56. Burggräf, P.; Dannapfel, M.; Wagner, J.; Heinbach, B.; Föhlisch, N.; Dackweiler, J., "ReLIFE": Business Models for Data-Based Remanufacturing: Adaptive Remanufacturing for Life Cycle Optimisation of Networked Capital Goods. In *The Monetization of Technical Data: Innovations from Industry and Research*, Springer: 2023; pp 507-520.
57. Barile, C.; Casavola, C.; Pappaletta, G.; Renna, G., Advancements in Electrospray Deposition (ESD) Technique: A Short Review. *Coatings* **2022**, *12* (10), 1536.
58. Widener, C. A.; Ozdemir, O. C.; Carter, M., Structural repair using cold spray technology for enhanced sustainability of high value assets. *Procedia Manufacturing* **2018**, *21*, 361-368.
59. Kattire, P.; Paul, S.; Singh, R.; Yan, W., Experimental characterization of laser cladding of CPM 9V on H13 tool steel for die repair applications. *Journal of Manufacturing Processes* **2015**, *20*, 492-499.
60. Megahed, S.; Koch, R.; Schleifenbaum, J. H., Laser Powder Bed Fusion Tool Repair: Statistical Analysis of 1.2343/H11 Tool Steel Process Parameters and Microstructural Analysis of the Repair Interface. *Journal of Manufacturing and Materials Processing* **2022**, *6* (6), 139.
61. Onuik, B.; Bandyopadhyay, A., Additive manufacturing in repair: Influence of processing parameters on properties of Inconel 718. *Materials Letters* **2019**, *252*, 256-259.
62. Jhavar, S.; Paul, C.; Jain, N., Causes of failure and repairing options for dies and molds: A review. *Engineering Failure Analysis* **2013**, *34*, 519-535.
63. Xu, Z.; Ouyang, W.; Jia, S.; Jiao, J.; Zhang, M.; Zhang, W., Cracks repairing by using laser additive and subtractive hybrid manufacturing technology. *Journal of Manufacturing Science and Engineering* **2020**, *142* (3), 031006.
64. Devoto, C.; Fernández, E.; Piñeyro, P., The economic lot-sizing problem with remanufacturing and inspection for grading heterogeneous returns. *Journal of Remanufacturing* **2021**, *11*, 71-87.
65. Zhang, M.; Amaitik, N.; Wang, Z.; Xu, Y.; Maisuradze, A.; Peschl, M.; Tzovaras, D., Predictive maintenance for remanufacturing based on hybrid-driven remaining useful life prediction. *Applied Sciences* **2022**, *12* (7), 3218.
66. Aras, N.; Boyaci, T.; Verter, V., The effect of categorizing returned products in remanufacturing. *IIE transactions* **2004**, *36* (4), 319-331.
67. Denizel, M.; Ferguson, M., Multiperiod remanufacturing planning with uncertain quality of inputs. *IEEE transactions on engineering management* **2009**, *57* (3), 394-404.
68. Sun, H.; Chen, W.; Liu, B.; Chen, X., Economic lot scheduling problem in a remanufacturing system with returns at different quality grades. *Journal of Cleaner Production* **2018**, *170*, 559-569.
69. Johnson, G., Ultrasonic flaw detectors... and beyond: a history of the discovery of the tools of nondestructive technology. *Quality* **2013**, *52* (4), 22-25.
70. Evangelista, D.; Terreran, M.; Pretto, A.; Moro, M.; Ferrari, C.; Menegatti, E. In *3D*

mapping of X-Ray images in inspections of aerospace parts, 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE: 2020; pp 1223-1226.

71. Cortina, M.; Arrizubieta, J. I.; Lamikiz, A.; Ukar, E., Impact of cutting fluid on hybrid manufacturing of AISI H13 tool steel. *Rapid Prototyping Journal* **2021**.

72. Du, W.; Bai, Q.; Wang, Y.; Zhang, B., Eddy current detection of subsurface defects for additive/subtractive hybrid manufacturing. *The International Journal of Advanced Manufacturing Technology* **2018**, 95, 3185-3195.

73. Jamshidi, M.; Chatelain, J.-F.; Rimpault, X.; Balazinski, M., Tool condition monitoring using machine tool spindle electric current and multiscale analysis while milling steel alloy. *Journal of Manufacturing and Materials Processing* **2022**, 6 (5), 115.

74. Altan, T.; Lilly, B.; Yen, Y., Manufacturing of dies and molds. *CiRP Annals* **2001**, 50 (2), 404-422.

75. Wanaguru, B. K.; Gamage, J. In *A Framework to Promote Mould Remanufacturing in Sri Lankan SME*, 2022 Moratuwa Engineering Research Conference (MERCon), IEEE: 2022; pp 1-6.

76. Li, Y.-j.; Li, H.; Lan, P.; Tang, H.-y.; Zhang, J.-q., Thermo-elasto-visco-plastic finite element analysis on formation and propagation of off-corner subsurface cracks in bloom continuous casting. *Journal of Iron and Steel Research International* **2017**, 24 (11), 1159-1168.

77. Ding, R.; Yang, H.; Li, S.; Hu, G.; Mo, J.; Chu, M.; Paddea, S.; Zhang, S.; Zhang, P.; Liu, Z., Failure analysis of H13 steel die for high pressure die casting Al alloy. *Engineering Failure Analysis* **2021**, 124, 105330.

78. Tsai, M.-Y.; Hsu, C. J.; Wang, C. O., Investigation of thermomechanical behaviors of flip chip BGA packages during manufacturing process and thermal cycling. *IEEE Transactions on Components and Packaging Technologies* **2004**, 27 (3), 568-576.

79. Ahmed, M.; Kamal, K.; Ratlamwala, T. A. H.; Hussain, G.; Alqahtani, M.; Alkahtani, M.; Alatefi, M.; Alzabidi, A., Tool Health Monitoring of a Milling Process Using Acoustic Emissions and a ResNet Deep Learning Model. *Sensors* **2023**, 23 (6), 3084.

80. Toshiyuki, B., Invitation to the Study of Dies and Molds. *Journal of International Economic Studies* **2017**, (31), 1-11.

81. Okada, A.; Okamoto, Y.; Clare, A.; Uno, Y., Fundamental study on releasability of molded rubber from mold tool surface. *The International Journal of Advanced Manufacturing Technology* **2014**, 70, 1515-1521.

82. Magliaro, J.; Cui, Z.; Shirzadian, S.; Green, D. E.; Altenhof, W.; Alpas, A. T., Evaluating die wear-induced edge quality degradation in trimmed DP980 steel sheets from in situ force response monitoring. *Wear* **2023**, 204792.

83. Um, J.; Rauch, M.; Hascoët, J.-Y.; Stroud, I., STEP-NC compliant process planning of additive manufacturing: remanufacturing. *The International Journal of Advanced Manufacturing Technology* **2017**, 88, 1215-1230.

84. Grzesik, W., Hybrid additive and subtractive manufacturing processes and systems: a review. *Journal of Machine Engineering* **2018**, 18 (4), 5-24.

85. Lee, J.; Kang, H.; Chu, W.; Ahn, S., Repair of damaged mold surface by cold-spray method. *CIRP annals* **2007**, 56 (1), 577-580.

86. De-Becker, D. The use of hybrid manufacturing in the repair and maintenance of railway lines. Loughborough University, 2020.

87. Lund, R. T.; Hauser, W. M., Remanufacturing-an American perspective. **2010**.

88. Sun, Q.; Shi, B.; Mu, X.; Sun, K., Fatigue strength evaluation for remanufacturing impeller of centrifugal compressor based on modified grey relational model. *Advances in Materials Science and Engineering* **2020**, 2020, 1-11.
89. Nasiri, S.; Khosravani, M. R., Applications of data-driven approaches in prediction of fatigue and fracture. *Materials Today Communications* **2022**, 33, 104437.
90. Nwankpa, C.; Eze, S.; Ijomah, W.; Gachagan, A.; Marshall, S., Achieving remanufacturing inspection using deep learning. *Journal of Remanufacturing* **2021**, 11, 89-105.
91. Mahmoud, D.; Magolon, M.; Boer, J.; Elbestawi, M.; Mohammadi, M. G., Applications of machine learning in process monitoring and controls of L-PBF additive manufacturing: a review. *Applied Sciences* **2021**, 11 (24), 11910.
92. Hawryluk, M.; Ziemba, J.; Zwierzchowski, M.; Janik, M., Analysis of a forging die wear by 3D reverse scanning combined with SEM and hardness tests. *Wear* **2021**, 476, 203749.
93. Xu, W.; Li, W.; Wang, Y., Experimental and theoretical analysis of wear mechanism in hot-forging die and optimal design of die geometry. *Wear* **2014**, 318 (1-2), 78-88.
94. Zhang, H.; Xu, C.; Xiao, D., Offline correction of tool path deviations for robot-assisted ultrasonic nondestructive testing. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* **2019**, 233 (8), 2879-2893.
95. Jiang, Z.; Grimm, M.; Zhou, M.; Hu, Y.; Esteban, J.; Navab, N., Automatic force-based probe positioning for precise robotic ultrasound acquisition. *IEEE Transactions on Industrial Electronics* **2020**, 68 (11), 11200-11211.
96. Honarvar, F.; Varvani-Farahani, A., A review of ultrasonic testing applications in additive manufacturing: Defect evaluation, material characterization, and process control. *Ultrasonics* **2020**, 108, 106227.
97. Kene, A. P.; Choudhury, S. K., Analytical modeling of tool health monitoring system using multiple sensor data fusion approach in hard machining. *Measurement* **2019**, 145, 118-129.
98. Novelo, X. E. A.; Chu, H.-Y., Application of vibration analysis using time-frequency analysis to detect and predict mechanical failure during the nut manufacturing process. *Advances in Mechanical Engineering* **2022**, 14 (2), 16878132221082758.
99. Shaloo, M.; Schnall, M.; Klein, T.; Huber, N.; Reitingner, B., A review of non-destructive testing (NDT) techniques for defect detection: application to fusion welding and future wire arc additive manufacturing processes. *Materials* **2022**, 15 (10), 3697.
100. Gupta, M.; Khan, M. A.; Butola, R.; Singari, R. M., Advances in applications of Non-Destructive Testing (NDT): A review. *Advances in Materials and Processing Technologies* **2022**, 8 (2), 2286-2307.
101. He, Y.; Li, M.; Meng, Z.; Chen, S.; Huang, S.; Hu, Y.; Zou, X., An overview of acoustic emission inspection and monitoring technology in the key components of renewable energy systems. *Mechanical Systems and Signal Processing* **2021**, 148, 107146.
102. Silva, M. I.; Malitckii, E.; Santos, T. G.; Vilaça, P., Review of conventional and advanced non-destructive testing techniques for detection and characterization of small-scale defects. *Progress in Materials Science* **2023**, 138, 101155.
103. Peter, W. T.; Wang, G. In *Sub-surface defects detection of by using active thermography and advanced image edge detection*, Journal of Physics: Conference Series, IOP Publishing: 2017; p 012029.
104. Tofail, S. A.; Mani, A.; Bauer, J.; Silien, C., In Situ, Real-Time Infrared (IR) Imaging for Metrology in Advanced Manufacturing. *Advanced Engineering Materials* **2018**, 20 (6), 1800061.

105. Zhong, S.; Nsengiyumva, W., Visual testing for fiber-reinforced composite materials. In *Nondestructive Testing and Evaluation of Fiber-Reinforced Composite Structures*, Springer: 2022; pp 97-132.
106. Ercetin, A.; Der, O.; Akkoyun, F.; Gowdru Chandrashekarappa, M. P.; Şener, R.; Çalışan, M.; Olgun, N.; Chate, G.; Bharath, K. N., Review of Image Processing Methods for Surface and Tool Condition Assessments in Machining. *Journal of Manufacturing and Materials Processing* **2024**, 8 (6), 244.
107. Farag, H., Eddy Current Probes Design for Defect Detection in Metallic Parts Made by Additive Manufacturing Processes. **2023**.
108. Schromm, T.; Beckmann, F.; Moosmann, J.; Berthe, D.; Pfeiffer, F.; Grosse, C., Challenges in non-destructive X-ray CT testing of riveted joints in the automotive industry. *Discover Applied Sciences* **2024**, 6 (7), 333.
109. Zeng, Y.; Wang, X.; Qin, X.; Hua, L.; Liu, G.; Guan, S., Laser ultrasonic inspection of defects in wire arc additive manufactured samples with different surface profiles. *Measurement* **2022**, 188, 110597.
110. Jadav, C.; Patel, S., Ultrasonic Testing For Determination Internal Flaws And Discontinuities In Metal Casting: A Review. *NVEO-NATURAL VOLATILES & ESSENTIAL OILS Journal| NVEO* **2021**, 4899-4918.
111. De Carvalho, C. C.; Inácio, P. L.; Miranda, R. M.; Santos, T. G., Using biotechnology to solve engineering problems: Non-destructive testing of microfabrication components. *Materials* **2017**, 10 (7), 788.
112. Klein, E., Some background history of ultrasonics. *The Journal of the Acoustical Society of America* **1948**, 20 (5), 601-604.
113. Bakar, S.; Ong, N.; Aziz, M.; Alcain, J.; Haimi, W.; Sauli, Z. In *Underwater detection by using ultrasonic sensor*, AIP Conference Proceedings, AIP Publishing LLC: 2017; p 020305.
114. Merz, E.; Abramowicz, J. S., 3D/4D ultrasound in prenatal diagnosis: is it time for routine use? *Clinical obstetrics and gynecology* **2012**, 55 (1), 336-351.
115. Workman, G. L.; Moore, P. O., *Nondestructive Testing Handbook: Ultrasonic Testing/Technical Editors: Gary L. Workman, Doron Kishon; Editor: Patrick O. Moore*. American Society for Nondestructive Testing: 2007.
116. Scholte, J., The range of existence of Rayleigh and Stoneley waves. *Geophysical Supplements to the Monthly Notices of the Royal Astronomical Society* **1947**, 5 (5), 120-126.
117. Burekhardt, C.; Grandchamp, P.-A.; Hoffmann, H., An experimental 2 MHz synthetic aperture sonar system intended for medical use. *IEEE Transactions on Sonics and Ultrasonics* **1974**, 21 (1), 1-6.
118. Holmes, C.; Drinkwater, B.; Wilcox, P., The post-processing of ultrasonic array data using the total focusing method. *Insight-Non-Destructive Testing and Condition Monitoring* **2004**, 46 (11), 677-680.
119. Wang, Z.; Ma, X.; Xu, Y. In *Weighted multi-element synthetic aperture focusing technique algorithm of ultrasonic non-destructive testing on machinery*, 2021 26th International Conference on Automation and Computing (ICAC), Portsmouth, United Kingdom, IEEE: Portsmouth, United Kingdom, 2021; pp 1-6.
120. Qin, K.; Yang, C.; Sun, F., Generalized frequency-domain synthetic aperture focusing technique for ultrasonic imaging of irregularly layered objects. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control* **2014**, 61 (1), 133-146.
121. Hossain, M. S.; Taheri, H.; Pudasaini, N.; Reichenbach, A.; Silwal, B. In *Ultrasonic nondestructive testing for in-line monitoring of wire-arc additive manufacturing (WAAM)*,

ASME International Mechanical Engineering Congress and Exposition, American Society of Mechanical Engineers: 2020; p V02BT02A037.

122. Bolotina, I. O.; Sednev, D. A.; Portenko, V. In *Ultrasonic testing method for quality control of mold castings*, IOP Conference Series: Materials Science and Engineering, IOP Publishing: 2019; p 012043.

123. Van Eck, N. J.; Waltman, L., VOSviewer manual. *Manual for VOSviewer version 2011*, 1 (0).

124. Godínez-Azcuaga, V. F.; Carlos, M. F.; Donahue, J. In *Automated ultrasonic testing during production: Avoiding bottlenecks and increasing throughput*, Proceedings of the 5th International Symposium on NDT in Aerospace, 2013; pp 1-8.

125. Khan, A.; Mineo, C.; Dobie, G.; Macleod, C.; Pierce, G., Vision guided robotic inspection for parts in manufacturing and remanufacturing industry. *Journal of Remanufacturing* **2021**, 11, 49-70.

126. Xiao, Z.; Xu, C.; Xiao, D.; Liu, F.; Yin, M., An optimized robotic scanning scheme for ultrasonic NDT of complex structures. *Experimental Techniques* **2017**, 41, 389-398.

127. Zimmermann, R.; Mohseni, E.; Wathavana Vithanage, R. K.; Lines, D.; MacLeod, C. N.; Pierce, G.; Gachagan, A.; Williams, S.; Ding, J.; Marinelli, G., Implementation of an ultrasonic total focusing method for inspection of unmachined wire+ arc additive manufacturing components through multiple interfaces. *Proceedings of the 47th Annual Review of Progress in Quantitative Nondestructive Evaluation, Virtual* **2020**, 25-26.

128. Zimmermann, R.; Mohseni, E.; Vithanage, R. K.; Lines, D.; Foster, E.; Macleod, C. N.; Pierce, S. G.; Marinelli, G.; Williams, S.; Ding, J., Increasing the speed of automated ultrasonic inspection of as-built additive manufacturing components by the adoption of virtual source aperture. *Materials & Design* **2022**, 220, 110822.

129. Mineo, C.; Cerniglia, D.; Poole, A., Autonomous robotic sensing for simultaneous geometric and volumetric inspection of free-form parts. *Journal of Intelligent & Robotic Systems* **2022**, 105 (3), 54.

130. Zimmermann, R.; Mohseni, E.; Lines, D.; Vithanage, R. K.; MacLeod, C. N.; Pierce, S. G.; Gachagan, A.; Javadi, Y.; Williams, S.; Ding, J., Multi-layer ultrasonic imaging of as-built wire+ Arc Additive manufactured components. *Additive Manufacturing* **2021**, 48, 102398.

131. Poole, A.; Sutcliffe, M.; Pierce, G.; Gachagan, A., A novel complete-surface-finding algorithm for online surface scanning with limited view sensors. *Sensors* **2021**, 21 (22), 7692.

132. Vicat, F., Automated Inspection of WIND BLADES Using a Collaborative Robot. *Quality* **2022**, 61 (4), 40-40.

133. Bakopoulou, K.; Michalos, G.; Mparis, K.; Gkournelos, C.; Dimitropoulos, N.; Makris, S., A Human Robot Collaborative Cell for automating NDT inspection processes. *Procedia CIRP* **2022**, 115, 214-219.

134. Sultan, T.; Dave, V. S.; Cetinkaya, C., Early Detection and Assessment of Invisible Cracks in Compressed Oral Solid Dosage Forms. *International Journal of Pharmaceutics* **2023**, 122786.

135. Ning, G.; Chen, J.; Zhang, X.; Liao, H., Force-guided autonomous robotic ultrasound scanning control method for soft uncertain environment. *International Journal of Computer Assisted Radiology and Surgery* **2021**, 16 (12), 2189-2199.

136. Lin, C.-Y.; Tran, C.-C.; Shah, S. H.; Ahmad, A. R., Real-Time Robot Pose Correction on Curved Surface Employing 6-Axis Force/Torque Sensor. *IEEE Access* **2022**, 10, 90149-90162.

137. Huang, Q.; Lan, J.; Li, X., Robotic arm based automatic ultrasound scanning for three-

- dimensional imaging. *IEEE Transactions on Industrial Informatics* **2018**, 15 (2), 1173-1182.
138. Abu-Dakka, F. J.; Saveriano, M., Variable impedance control and learning—a review. *Frontiers in Robotics and AI* **2020**, 7, 590681.
139. Park, J.; Goswami, A.; Vadakkepat, P., Compliance/impedance control strategy for humanoids. *Humanoid Robotics: A Reference* **2019**, 1009-1028.
140. Kadalagere Sampath, S.; Wang, N.; Wu, H.; Yang, C., Review on human-like robot manipulation using dexterous hands. *Cognitive Computation and Systems* **2023**.
141. Deng, Y.; Wang, G.; Yue, X.; Zhou, K. In *A Review of Robot Grinding and Polishing Force Control Mode*, 2022 IEEE International Conference on Mechatronics and Automation (ICMA), IEEE: 2022; pp 1413-1418.
142. Gold, T.; Völz, A.; Graichen, K., Model predictive interaction control for robotic manipulation tasks. *IEEE Transactions on Robotics* **2022**, 39 (1), 76-89.
143. Anand, A. S.; Gravdahl, J. T.; Abu-Dakka, F. J., Model-based variable impedance learning control for robotic manipulation. *Robotics and Autonomous Systems* **2023**, 170, 104531.
144. Szabo, R., Developing Different Test Conditions to Verify the Robustness and Versatility of Robotic Arms Controlled by Evolutionary Algorithms. *Electronics* **2024**, 13 (11), 2130.
145. Li, L.; Huang, T.; Pan, C.; Pan, J.; Su, W., Impedance control for force tracking of a dual-arm cooperative robot based on particle swarm optimization. *Industrial Robot: the international journal of robotics research and application* **2024**, 51 (3), 436-445.
146. Wu, G.; Zhang, P.; Zhang, J.; Ma, A.; Xu, W. In *End-Effector Impedance Control of Robotic Arm Based on Enhanced Neural Network RBF-PID-PSO*, 2023 2nd International Conference on Artificial Intelligence, Human-Computer Interaction and Robotics (AIHCIR), IEEE: 2023; pp 357-364.
147. Li, J.; Shi, H.; Hwang, K.-S., Using goal-conditioned reinforcement learning with deep imitation to control robot arm in flexible flat cable assembly task. *IEEE Transactions on Automation Science and Engineering* **2023**.
148. Surindra, M. D.; Alfariy, G. A. F.; Caesarendra, W.; Petra, M. I.; Prasetyo, T.; Tjahjowidodo, T.; Królczyk, G. M.; Glowacz, A.; Gupta, M. K., Use of machine learning models in condition monitoring of abrasive belt in robotic arm grinding process. *Journal of Intelligent Manufacturing* **2024**, 1-14.
149. Gibbs, A.; Scott, T.; Gonzalez, C.; Barbosa, R.; Coro, R.; Dizor, R.; McCrory, S.; Regez, B.; Rahman, S. M. In *Impedance-Based Feedforward Learning Control for Natural Interaction between a Prosthetic Hand and the Environment*, 2021 IEEE 2nd International Conference on Human-Machine Systems (ICHMS), IEEE: 2021; pp 1-4.
150. Jiang, S.; Wong, L. L. In *Active Tactile Exploration using Shape-Dependent Reinforcement Learning*, 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE: 2022; pp 8995-9002.
151. Zhang, T.; Xiao, M.; Zou, Y.-b.; Xiao, J.-d.; Chen, S.-y., Robotic curved surface tracking with a neural network for angle identification and constant force control based on reinforcement learning. *International Journal of Precision Engineering and Manufacturing* **2020**, 21, 869-882.
152. Wang, F.; Cui, B.; Liu, Y.; Ren, B., Deep Reinforcement Learning for Peg-in-hole Assembly Task Via Information Utilization Method. *Journal of Intelligent & Robotic Systems* **2022**, 106 (1), 16.
153. Zachares, P.; Heravi, N., Uncertainty Quantification of a Classification Network for Contact-Rich Manipulation.

154. Jiono, M.; Lin, H.-I. In *Compliant Control using Force Sensor for Industrial Robot*, 2024 10th International Conference on Mechatronics and Robotics Engineering (ICMRE), IEEE: 2024; pp 51-55.
155. Shahid, S. T.; Siddique, S. M. A.; Bhuiyan, M. H. K., Automatic Contact-Based 3D Scanning Using Articulated Robotic Arm. *arXiv preprint arXiv:2411.07047* **2024**.
156. Li, D.; Yang, J.; Zhao, H.; Ding, H., Contact force plan and control of robotic grinding towards ensuring contour accuracy of curved surfaces. *International Journal of Mechanical Sciences* **2022**, 227, 107449.
157. Raina, D.; Mathur, A.; Voyles, R. M.; Wachs, J.; Chandrashekhara, S. H.; Saha, S. K. In *Rusopt: Robotic ultrasound probe normalization with bayesian optimization for in-plane and out-plane scanning*, 2023 IEEE 19th International Conference on Automation Science and Engineering (CASE), IEEE: 2023; pp 1-7.
158. Droste, R.; Drukker, L.; Papageorgiou, A. T.; Noble, J. A. In *Automatic probe movement guidance for freehand obstetric ultrasound*, Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part III 23, Springer: 2020; pp 583-592.
159. Malczyk, G.; Brunner, M.; Cuniato, E.; Tognon, M.; Siegwart, R., Multi-directional interaction force control with an aerial manipulator under external disturbances. *Autonomous Robots* **2023**, 47 (8), 1325-1343.
160. Yue, P.; Xin, J.; Zhao, H.; Liu, D.; Shan, M.; Zhang, J. In *Experimental research on deep reinforcement learning in autonomous navigation of mobile robot*, 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA), IEEE: 2019; pp 1612-1616.
161. Xue, X.; Li, Z.; Zhang, D.; Yan, Y. In *A deep reinforcement learning method for mobile robot collision avoidance based on double dqn*, 2019 IEEE 28th International Symposium on Industrial Electronics (ISIE), IEEE: 2019; pp 2131-2136.
162. Zhao, W.; Queralta, J. P.; Qingqing, L.; Westerlund, T. In *Towards closing the sim-to-real gap in collaborative multi-robot deep reinforcement learning*, 2020 5th International Conference on Robotics and Automation Engineering (ICRAE), IEEE: 2020; pp 7-12.
163. Tsardoulis, E.; Mitkas, P., Robotic frameworks, architectures and middleware comparison. *arXiv preprint arXiv:1711.06842* **2017**.
164. Al-Geddawy, T., A digital twin creation method for an opensource low-cost changeable learning factory. *Procedia Manufacturing* **2020**, 51, 1799-1805.
165. Körber, M.; Lange, J.; Rediske, S.; Steinmann, S.; Glück, R., Comparing popular simulation environments in the scope of robotics and reinforcement learning. *arXiv preprint arXiv:2103.04616* **2021**.
166. De Melo, M. S. P.; da Silva Neto, J. G.; Da Silva, P. J. L.; Teixeira, J. M. X. N.; Teichrieb, V. In *Analysis and comparison of robotics 3d simulators*, 2019 21st Symposium on Virtual and Augmented Reality (SVR), IEEE: 2019; pp 242-251.
167. Collins, J.; Chand, S.; Vanderkop, A.; Howard, D., A review of physics simulators for robotic applications. *IEEE Access* **2021**, 9, 51416-51431.
168. Beyret, B.; Shafti, A.; Faisal, A. A. In *Dot-to-dot: Explainable hierarchical reinforcement learning for robotic manipulation*, 2019 IEEE/RSJ International Conference on intelligent robots and systems (IROS), IEEE: 2019; pp 5014-5019.
169. Hebecker, M.; Lambrecht, J.; Schmitz, M. In *Towards real-world force-sensitive robotic assembly through deep reinforcement learning in simulations*, 2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), IEEE: 2021; pp 1045-1051.
170. Yang, X.; Ji, Z.; Wu, J.; Lai, Y.-K. In *An open-source multi-goal reinforcement learning*

environment for robotic manipulation with pybullet, Towards Autonomous Robotic Systems: 22nd Annual Conference, TAROS 2021, Lincoln, UK, September 8–10, 2021, Proceedings, Springer: 2021; pp 14-24.

171. Liu, G.; De Winter, J.; Vanderborght, B.; Nowé, A.; Steckelmacher, D. In *MoveRL: To a Safer Robotic Reinforcement Learning Environment*, Artificial Intelligence and Machine Learning: 33rd Benelux Conference on Artificial Intelligence, BNAIC/Benelearn 2021, Esch-sur-Alzette, Luxembourg, November 10–12, 2021, Revised Selected Papers 33, Springer: 2022; pp 239-253.

172. Aiello, A. Robotic arm pick-and-place tasks: Implementation and comparison of approaches with and without machine learning (deep reinforcement learning) techniques. Politecnico di Torino, 2020.

173. Abdi, A.; Ranjbar, M. H.; Park, J. H., Computer vision-based path planning for robot arms in three-dimensional workspaces using Q-learning and neural networks. *Sensors* **2022**, *22* (5), 1697.

174. Yu, L.; Xia, Y.; Wang, P.; Sun, L., Automatic adjustment of laparoscopic pose using deep reinforcement learning. *Mechanical Sciences* **2022**, *13* (1), 593-602.

175. Villalobos, J.; Sanchez, I. Y.; Martell, F., Singularity analysis and complete methods to compute the inverse kinematics for a 6-DOF UR/TM-Type robot. *Robotics* **2022**, *11* (6), 137.

176. O'Kane, J. M., A gentle introduction to ROS. **2014**.

177. Wang, Z.; Zhang, M.; Xu, Y. In *Development of a Robotic Arm Control Platform for Ultrasonic Testing Inspection in Remanufacturing*, 2022 27th International Conference on Automation and Computing (ICAC), IEEE: 2022; pp 1-6.

178. UniversalRobots Universal_Robots_ROS_Driver.
https://github.com/UniversalRobots/Universal_Robots_ROS_Driver.

179. MetaMemory Robotiq_85_gripper. https://github.com/artursg/robotiq_85_gripper.git.

180. JenniferBuehler, gazebo-pkgs. **2021**.

181. Scherzinger, S.; Roennau, A.; Dillmann, R. In *Forward Dynamics Compliance Control (FDCC): A new approach to cartesian compliance for robotic manipulators*, 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada, IEEE: Vancouver, Canada, 2017; pp 4568-4575.

182. Lin, J. C.; Lin, W.; Lee, K.; Tong, J., The optimal clearance design of micro-punching die. *Journal of Achievements in Materials and Manufacturing Engineering* **2008**, *29* (1), 79-82.

183. Liu, R.; Wan, W.; Koyama, K.; Harada, K., Multi-Pen Robust Robotic 3D Drawing Using Closed-Loop Planning. *arXiv preprint arXiv:2009.14501* **2020**.

184. Mahdi, S. M.; Raafat, S. M., Robust Interactive PID Controller Design for Medical Robot System. *International Journal of Intelligent Engineering & Systems* **2022**, *15* (1).

185. Zhang, B.; Wu, S.; Wang, D.; Yang, S.; Jiang, F.; Li, C., A review of surface quality control technology for robotic abrasive belt grinding of aero-engine blades. *Measurement* **2023**, 113381.

186. Gracia, L.; Solanes, J. E.; Muñoz-Benavent, P.; Miro, J. V.; Perez-Vidal, C.; Tornero, J. In *A sliding mode control architecture for human-manipulator cooperative surface treatment tasks*, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE: 2018; pp 1318-1325.

187. Zhao, D.; Guo, W., Shape and performance controlled advanced design for additive manufacturing: a review of slicing and path planning. *Journal of Manufacturing Science and Engineering* **2020**, *142* (1), 010801.

188. Ning, G.; Zhang, X.; Liao, H., Autonomic robotic ultrasound imaging system based on reinforcement learning. *IEEE Transactions on Biomedical Engineering* **2021**, 68 (9), 2787-2797.
189. Ezquerro, A.; Rodriguez, M. A.; Tellez, R. openai_ros. http://wiki.ros.org/openai_ros.
190. Kapukotuwa, J.; Lee, B.; Devine, D.; Qiao, Y. In *MultiROS: ROS-Based Robot Simulation Environment for Concurrent Deep Reinforcement Learning*, 2022 IEEE 18th International Conference on Automation Science and Engineering (CASE), IEEE: 2022; pp 1098-1103.
191. Jordan, S.; Chandak, Y.; Cohen, D.; Zhang, M.; Thomas, P. In *Evaluating the performance of reinforcement learning algorithms*, International Conference on Machine Learning, PMLR: 2020; pp 4962-4973.
192. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O., Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* **2017**.
193. Mnih, V., Asynchronous Methods for Deep Reinforcement Learning. *arXiv preprint arXiv:1602.01783* **2016**.
194. Fang, H.; Tu, Y.; Wang, H.; He, S.; Liu, F.; Ding, Z.; Cheng, S. S., Fuzzy-Based adaptive optimization of unknown Discrete-Time Nonlinear Markov jump systems with Off-Policy reinforcement learning. *IEEE Transactions on Fuzzy Systems* **2022**, 30 (12), 5276-5290.
195. Wang, N.; Zhou, J.; Zhang, X. In *Research on the Estimation of Sensor Bias and Parameters of Load Based on Force-Feedback*, International Conference on Intelligent Robotics and Applications, Springer: 2018; pp 404-413.
196. Ju, H.; Juan, R.; Gomez, R.; Nakamura, K.; Li, G., Transferring policy of deep reinforcement learning from simulation to reality for robotics. *Nature Machine Intelligence* **2022**, 1-11.
197. Hanna, J. P.; Desai, S.; Karnan, H.; Warnell, G.; Stone, P., Grounded action transformation for sim-to-real reinforcement learning. *Machine Learning* **2021**, 110 (9), 2469-2499.
198. Bogunowicz, D.; Rybníkov, A.; Vendidandi, K.; Chervinskii, F., Sim2real for peg-hole insertion with eye-in-hand camera. *arXiv preprint arXiv:2005.14401* **2020**.
199. Orsula, A.; Bøgh, S.; Olivares-Mendez, M.; Martinez, C. In *Learning to Grasp on the Moon from 3D Octree Observations with Deep Reinforcement Learning*, 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE: 2022; pp 4112-4119.
200. Rusu, A. A.; Večerík, M.; Rothörl, T.; Heess, N.; Pascanu, R.; Hadsell, R. In *Sim-to-real robot learning from pixels with progressive nets*, Conference on robot learning, PMLR: 2017; pp 262-270.
201. Tzeng, E.; Devin, C.; Hoffman, J.; Finn, C.; Abbeel, P.; Levine, S.; Saenko, K.; Darrell, T. In *Adapting deep visuomotor representations with weak pairwise constraints*, Algorithmic Foundations of Robotics XII: Proceedings of the Twelfth Workshop on the Algorithmic Foundations of Robotics, Springer: 2020; pp 688-703.
202. Arndt, K.; Hazara, M.; Ghadirzadeh, A.; Kyrki, V. In *Meta reinforcement learning for sim-to-real domain adaptation*, 2020 IEEE international conference on robotics and automation (ICRA), IEEE: 2020; pp 2725-2731.
203. Szot, A.; Clegg, A.; Undersander, E.; Wijmans, E.; Zhao, Y.; Turner, J.; Maestre, N.; Mukadam, M.; Chaplot, D. S.; Maksymets, O., Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems* **2021**, 34, 251-266.
204. Xia, F.; Zamir, A. R.; He, Z.; Sax, A.; Malik, J.; Savarese, S. In *Gibson env: Real-world*

perception for embodied agents, Proceedings of the IEEE conference on computer vision and pattern recognition, 2018; pp 9068-9079.

Appendix A: Implementation of ROS

ROS is an open-source middleware, which was launched by Stanford University PhD students Eric Berger and Keenan Wyronek in 2007. The original target of ROS is to develop a middleware that is suitable for everyone with no deep robotic knowledge foundations. With the development through these years, ROS has become not only a middleware, but also a community of robotic researchers and developers, for example, there have been over 150,000 repositories for ROS on the largest software development platform, Github.

ROS is currently releasing ROS2, which is more supportive on real-time control, however, there are some stability issues on ROS2, therefore ROS1 is still in use in this study. Even though ROS is compatible to Windows operating system, it is preferred that Linux system to be used with ROS. Since ROS is designed for Linux kernel and more packages and codes have been tested in Linux environment, but the main tools, such as catkin tools are not compatible in Windows OS. To use updated python 3 libraries, Ubuntu 20.04 system is used in this study. Accordingly, ROS Noetic is used on Ubuntu 20.04 (see Figure 117).



Figure 117 Top: “nine dots” ROS logo. Bottom left: logo of ROS Noetic Ninjemys, bottom right: screenshot of Ubuntu 20.04 system.

Since there will be some problems during building dependencies of ROS due to dependencies incompatible issues or installation issues, Miniconda is used along with ROS to secure installation of necessary dependencies. Miniconda is a minimal installer for Conda, a package and environment management system. It allows developers to create and manage isolated environments with different versions of Python and other packages. It has been proven that Conda helped build of ROS a lot.

The first step is to build a conda environment, an environment with python version 3.7 was created in this study. After creating of new environment, the conda environment needs to be activated each time when ROS is launched (see Figure 118). As shown in the figure, if it shows “(base)” before the command line in the terminal, it indicates that the conda environment is working. A customised conda environment needs to be activated and source to the setup.bash. It contains environment variable settings and configurations needed for ROS packages to work correctly in the workspace. Sourcing this file ensures that ROS commands and packages are available in the current shell session. After this conda environment establishment, ROS is installed in Ubuntu.

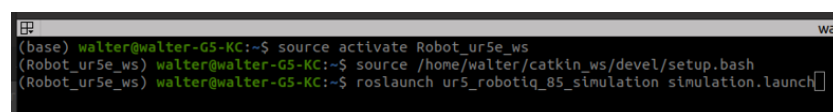


Figure 118 Base environment and created Robot_ur5e_ws environment in Conda.

ROS is a middleware which is mainly based on C++ and Python. Many core components and libraries of ROS are implemented in C++. This includes the ROS middleware, communication libraries, and key functionalities. To build the inter-dependent system, ROS community built the tool, catkin tools. Catkin is the build system that ROS uses to manage the compilation and build process of ROS packages. It organizes code into packages and provides tools for building, testing, and managing dependencies. Catkin uses CMake as its underlying build system, allowing for a more modular and flexible build process. Catkin organizes code into packages, each with its own CMakeLists.txt file. An advantage of this approach is that the total configuration would be smaller than configuring each package individually and that the Make targets can be parallelized even among dependent packages. Catkin Tools provides better dependency management compared to the older rosbuilt system. Every time after significant change including the first-time installation has been made in ROS, a “catkin_make” in the workspace of ROS is necessary. ROS also has extensive support for Python. Python is commonly used for writing high-level scripts, prototyping, and interacting with various ROS functionalities. One of the strengths of ROS is its support for interoperability between components written in different languages. ROS nodes written in C++ and Python can communicate seamlessly through the ROS middleware.

ROS is basically a system based on information threads. It helps developers build and manage software for robotic systems. Despite its name, ROS is not an operating system in the traditional sense. Instead, it provides a set of tools and libraries that facilitate communication and coordination among the various components of a robotic system. The main format of document ran by operators in ROS is the *.launch* file, which is an XML (eXtensible Markup Language) format text document (see Figure 122). In each *.launch* file, it contains the information of a sender, a receiver, a heading and a message body. Different from HTML files, XML files are designed to transfer data which is suitable for ROS structure. In the launch file, it defined the environment that the robot will run, which is *.world* file. The default world used in Gazebo is “empty_world”. Within this world file, the illuminations and gravity properties within the environment can be defined in the *.world* file. Within the world, there are model files, Unified Robot Description Format (urdf) and Simulation Description Format (sdf), which are the objects in the simulation environment. Any objects in normal world can be defined if there is a 3D model of the object, e.g., a drink can or a vehicle. The mass, inertia, material, texture and mesh can be defined in the text-based document. The robot model needs to be “spawned” when starting the *launch* file. The model of robot is in the format of *.urdf* format, in which every joint and link is defined. Links should be inserted in urdf as *.dae* format, and described in text for other features. Joint is the connection between two main parts of the robotic arm, it is normally rotational in robotic case. The link is the “main part” of robot, in UR5e case, 6 joints and 7 links are defined (shown in Figure 119). Except for the text description of these links and joints, the 3D model is also needed for further simulation. The links in the urdf file are connected with the relation as parent link and child link, and with the relationship, the transform tree (TF tree) will be established in ROS to realise dynamic pose/position transform (see Figure 120).

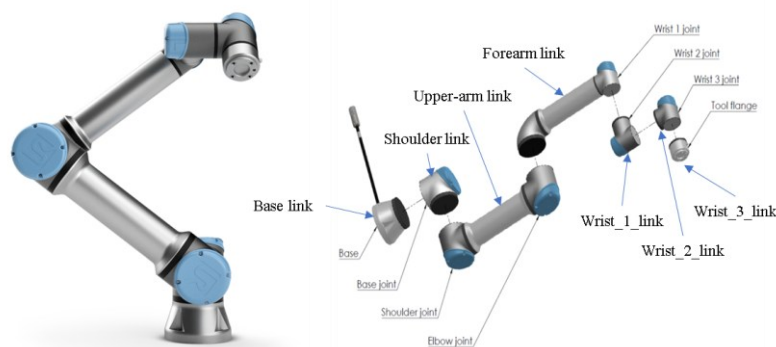


Figure 119 Joints and links of UR5e robot.

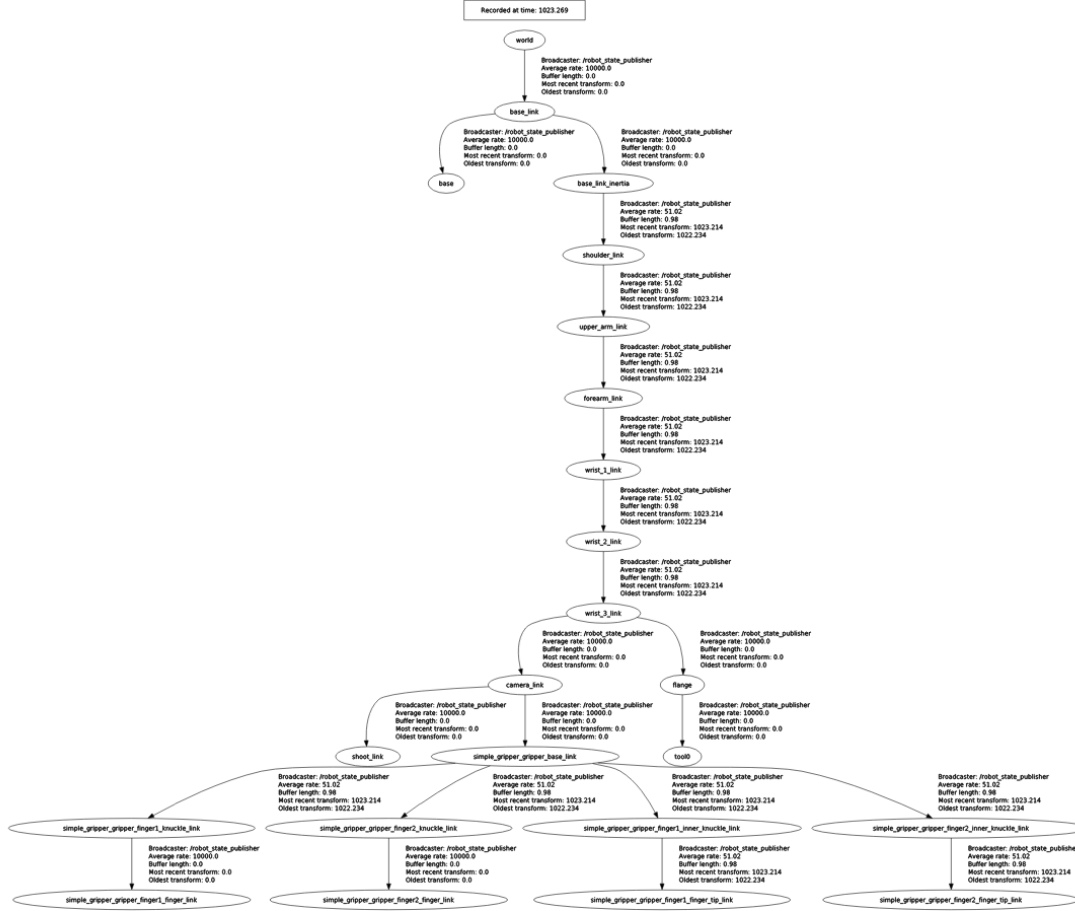


Figure 120 Example of TF tree of the robot.

The *launch* file also includes the parameters of the ROS running. There are two kinds of parameter settings, *param* and *rosparam*. *Param* can be used to set single parameter in ROS, it can be set within the *launch* file. While *rosparam* is to load the parameters set up in the *.yaml* (yet another markup language) files. These parameters are specified for the robot, they are set in a separate file since the parameters are more complex. *Group* is the tag to set up a group of parameters within the *launch* file. Within the group, *arg* is to define a single argument, which can be only used within the *launch* file. The value of the argument can be set in text, it can be also set as a 'default' value and overwritten by command lines in the future commands. *Include* is to set up the set-up files needed to be included, such as *world* files, *launch* files or python files. The screenshot of a launch file can be seen in Figure 121. As can be seen in the figure, the robot description is cited in line 15, which has been defined in line 6, pointed to the directory of the robot description.

```

1 <?xml version="1.0"?>
2 <launch>
3   <arg name="transmission_hw_interface" default="hardware_interface/PositionJointInterface" />
4
5   <!-- send robot urdf to param server -->
6   <param name="robot_description"
7     command="$(find xacro)/xacro --inorder '$(find
8       ur5robotiq_moveit_config)/urdf/ur5_robotiq_macro.xacro'
9       transmission_hw_interface:=$(arg transmission_hw_interface)" />
10
11   <group if="$(arg start_gazebo)">
12     <include file="$(find gazebo_ros)/launch/empty_world.launch">launch empty world again, can be another
13     world launch file.
14   </include>
15   </group>
16
17   <node name="spawn_urdf" pkg="gazebo_ros" type="spawn_model" args="-x 1 -y 1 -z 1 -param
18     robot_description -urdf -model ur5" />
19
20   <!-- Load joint controller configurations from YAML file to parameter server -->
21   <rosparam file="/home/walter/catkin_ws/src/ur5_robotiq_85_simulation/config/ur5_robotiq_85_control.yaml"
22     command="load" />
23   /home/walter/catkin_ws/src/ur5robotiq_moveit_config/config/ros_controllers.yaml
24 </launch>

```

Figure 121 Screenshot of a launch file.

Unified robotic description format (URDF) or simulation description format (SDF) file is the robot description file which defined the structure of the robot by text. Within the description file, the inertia, mass, material, size and friction features of the links will be defined. Besides the description files of the robot, mesh files (.dae or .stl) and config files are necessary for the simulation purpose. Dae file is used in COLLADA, which stands for "Collaborative Design Activity". Stl (stereolithography) file is used in most 3D CAD software, such as, Solidworks, Meshlab. Robot links are normally defined in .dae format and cited in .urdf format. Other objects in simulation are normally defined in .stl format and cited in .sdf files and used in simulation. However, there will be special cases, such as, when plugging objects, such as the holder, into the robot model, the .stl file will be inserted into the .urdf file. Materials and plugins can be also defined. In plug-in, sensors, such as cameras, inertial measurement unit (IMU) can be defined and installed on the robot. Subsequently, the setting of the robot model can be visualised in Gazebo or RViz.

The ROS works based on packages, each package communicates with each other by sending and receiving messages. The package can be created based on *roscpp* or *rospy*, which are "ROS+ C++" and "ROS+ Python". *Roscpp* is the implementation of ROS on C++, it is a pre-defined client library which can quickly interact with topics, services and parameters. ROS is designed based on *roscpp*, it is used to make high-performance packages. Similarly, *rospy* is the similar client library for Python. The advantage of *rospy* is the speed of implementation. After the spawn of robotic arm, related *roscpp* will be established. From each *roscpp*, there will be some arguments within the node, and there will be *rostopic* generated from the nodes. ROS master will take charge of how the nodes are named, and the master can be invoked by using *roscpp* command. Besides the spawn of the robotic arm model, the controllers of the robotic arm and the gripper will be spawned and the *roscpp* for the controllers will be established. The other *roscpp*, such as for the Moveit motion planning, publisher of the joint positions, are also started when the .launch file starts. The messages can be delivered between *roscpp* via *rostopic*. The *rostopic* are established as soon as the publisher or the subscriber of a topic is ready within a *roscpp*. The message, or the data will be delivered via the *rostopic*. The *rostopic* are mostly defined in the library and easy to use. Use command in the prompt window, the information of the *rostopic* can be checked. For some nodes, such as *spawn_model*, *rosservice* are used to finish the node. *Roservice* can also deliver messages. Unlike *rostopic*, *rosservice* are bi-direction one-to-one communication. It invokes request to one node and receives response from the node. For each *roscpp*, "pkg", "type", "respawn", "output" and "args" should be defined. The pkg is the defined *roscpp* name, which is in the directory of the *roscpp*. When creating the ROS package, the core code has been created to create the *roscpp*. In the "package.xml" file of each package, the dependant files are defined in the file to align the *roscpp* "pkg" to defined files. The *roscpp* should be connected to the "catkin" libraries. "catkin" is a build system of ROS to generate the target packages from raw source code to the end users. The products of catkin can be libraries, executable programs, simulations, or anything else than static codes. Catkin in ROS is similar to CMake, which is a

common build tool for C++. Therefore, in each package of ROS, there is a CMakeList.txt, which is defining the processes of preparing and executing the build process.

Besides the physical settings in ROS, the controller of the robot should be also designed. ROS has application programming interface (API) to main coding languages, i.e., python and C++. The controllers define how the joints and links move under certain condition, for example, position controller will guarantee the joints and the links to move to the targeted positions, with the help of ROS API library that has been defined by ROS developers, such as, moveit_commander transfers the defined rosnode in Moveit, e.g., the position of the end-effector to the coding. In this thesis, the ROS platform was established to realise these functions: simulation, implementation of surface scanning tasks, implementing reinforcement learning algorithms, synchronisation of simulation models and real robot, control of real robots. The communication between ROS and the real robot can be realised via Ethernet cable. The main structure of ROS is listed in Figure 122.

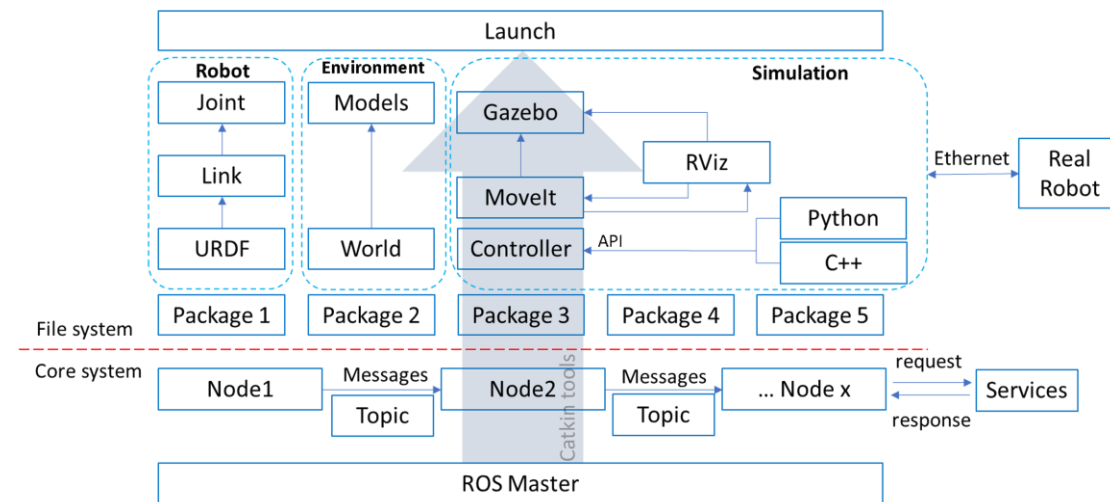


Figure 122 Main structure of ROS.

Ros_control is a package that takes the joints states as input and calculate the effort output to realise the control closed loop (shown in Figure 123). The information of the joints, such as position, speed or effort is sent to ros_control as input. The calculated output will be sent to the actuators in the robot. Ros_control contains controller manager, controller interface, transmissions, hardware interfaces. In controller manager, it manages the start, stop, load and unload of the controllers. Hardware interface is for communication with real hardware. To make the ros_control in Gazebo simulation, the “transmission” from normal joints to actuators should be made in the urdf file. Plug-in is also needed when using ros_control in Gazebo. This plug-in is specially for parsing the urdf and loading the appropriate hardware interfaces in Gazebo. In ros_control, the joint position control normally uses the proportional integral derivative (PID) control. The parameters of PID should be set up in the .yaml config file. The commands to the joints will be sent to the ROS interface, i.e., joint_states. Besides the control of joints, the planner is also needed. A classic planner for ROS is Moveit.

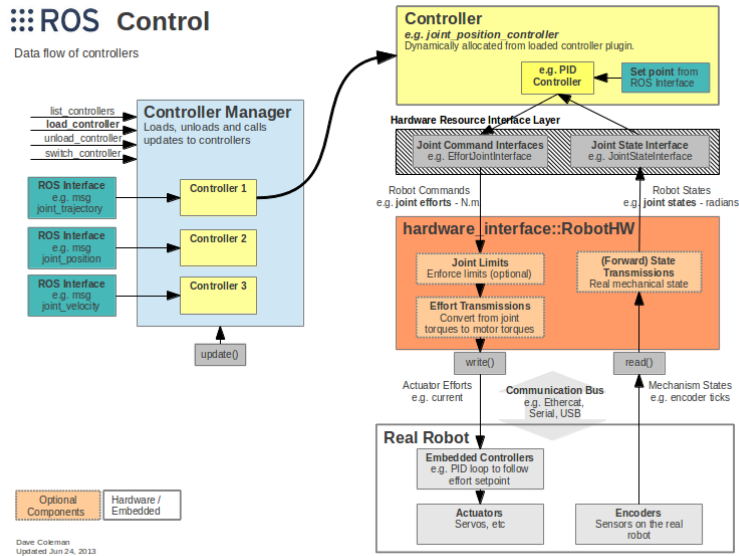


Figure 123 Structure of ROS_control (Source: ROS wiki).

Parameter server is a component of ROS, which can contain the configuration data. It is a centralised storage that is accessed by multiple nodes. Configuration data, such as, robot geometry, sensor calibration, controllers, and other parameters are saved here. The parameters can be saved to the parameter server by any node, and can be accessed by other nodes which have the permission. Any parameter in the launch file, which starts with “*param name*” will be saved to the parameter server. With the command “rosparam list” in the terminal window, all the parameters on the server can be checked.

Moveit is a software that can define the movement of the robot. It is an embedded solution for the inverse kinematic problem. To start setting up Moveit, command in Linux is “roslaunch moveit_setup_assistant setup_assistant.launch”, which will start the setup assistant of Moveit (shown in Figure 124). At first, the name list of joints and links that has been defined by the developers will be listed. A control group will be established by selecting related links, for example, 7 links for UR5e. After this, a move group will be established with all these links. Subsequently, abnormal collision and self-collision will be checked. Certain positions of the joints can be defined as particular positions of the robot and the gripper. Controllers of robot and gripper are also set up in Moveit. Moveit has several defined inverse solvers that can be used to solve the inverse kinematic problem. The targeted positions can be assigned to the robot, the solver will calculate the precise positions of the joints, with an optimal solution. Many classic planners in OMPL (Open Motion Planning Library) Planners are provided in Moveit, such as, RRT (Rapidly Exploring Random Tree), RRTConnect, etc. Moveit is a mature solution for reverse kinematic problem, even though sometimes it gives non-available solutions, but most of the time it is working, and it saves time for kinematic problem solution.



Figure 124 Screenshot of Moveit setup.

After the MoveIt setup, the final step of the MoveIt assistant is to generate the packages, which is normally named as “*_moveit_config”. Within this package, there will be originally three files: CMakeLists.txt, package.xml, .setup_assistant and two folders: config, launch. In the MoveIt framework, the CMakeLists.txt and package.xml files in the moveit_config folder are essential for configuring and building MoveIt configurations. The CMakeLists.txt file is written in the CMake scripting language and is used by CMake, the build system used by ROS, to control the software build process. In the context of MoveIt and moveit_config, the CMakeLists.txt file typically includes instructions for building and configuring the MoveIt configuration package. It specifies the minimum required version of CMake and other dependencies. It helps define the source files to be compiled, executable targets, and any additional build instructions. It helps specify ROS-specific information, such as dependencies on other ROS packages, ROS package export settings, and ROS messages/services. It configures MoveIt plugins and extensions that are specific to the MoveIt configuration package. For the package.xml, it is an XML file used to declare metadata about the ROS package. It provides information about the package name, version, maintainer, license, and dependencies on other ROS packages. And for the config folder, it contains mainly the controllers codes. In launch folder, most of the original roslaunch files are located there, such as, demo_gazebo.launch.

The high-end 3D printing machine Ultimaker S5 was used to manufacture it with polylactic acid (PLA) material (shown in Figure 125). PLA is a commonly used material for industrial application. PLA material has some advantages, such as, relatively acceptable price, good strength and elastic, low thermal expansion, good layer adhesion. PLA will degrade when temperature is over 60°C, which is not possible in this application. The good output performance of 3D printing is secured by adjusting the printing parameters, such as, feeding speed, temperature. The optimal extruding temperature of the filament should be 190-220 °C. Since the target contact force between the transducer and the surface should not beyond 20N, and all the experiment processes are monitored, so the strength of PLA material is enough for this study.

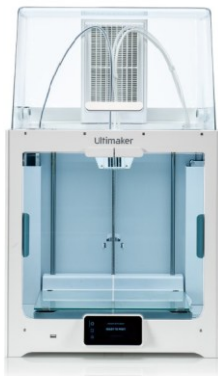
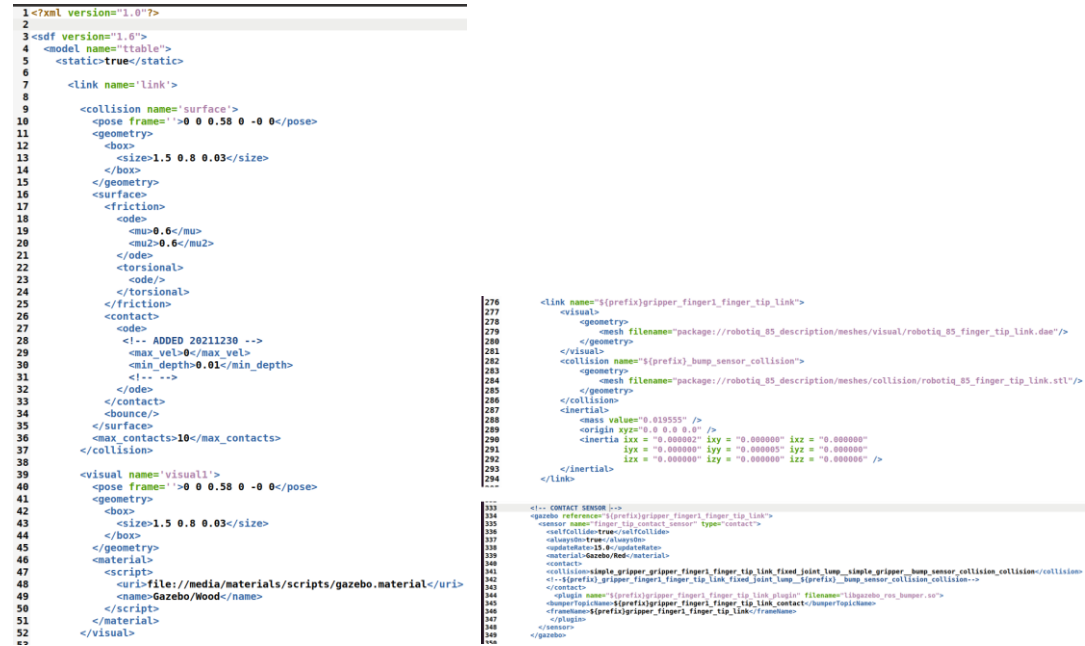


Figure 125 Ultimaker S5 3D printer used in this study.

Appendix B: Codes in This Study



```

1 <?xml version="1.0"?>
2
3 <sdf version="1.6">
4   <model name="tttable">
5     <static true</static>
6
7     <link name="link">
8
9       <collision name="surface">
10        <pose frame="1">0 0 0.58 0 -0 0</pose>
11        <geometry>
12          <box>
13            <size>1.5 0.8 0.03</size>
14          </box>
15        </geometry>
16        <surface>
17          <friction>
18            <ode>
19              <mu>0.6</mu>
20              <mu2>0.6</mu2>
21            </ode>
22            <torsional>
23              <ode>
24                </torsional>
25              </ode>
26            </friction>
27          </surface>
28        </collision>
29        <contact>
30          <ode>
31            <!-- ADDED 20211230 -->
32            <max_vel>0</max_vel>
33            <min_depth>0.01</min_depth>
34            <!-- -->
35          </ode>
36        </contact>
37      </link>
38
39      <visual name="visual1">
40        <pose frame="1">0 0 0.58 0 -0 0</pose>
41        <geometry>
42          <box>
43            <size>1.5 0.8 0.03</size>
44          </box>
45        </geometry>
46        <material>
47          <script>
48            <uri>file://media/materials/scripts/gazebo.material</uri>
49            <name>Gazebo/Wood</name>
50          </script>
51        </material>
52      </visual>
53
276   <link name="$(prefix)gripper_finger1_finger_tip_link">
277     <visual>
278       <geometry>
279         <mesh filename="package://robotiq_85_description/meshes/visual/robotiq_85_finger_tip_link.dae">
280       </geometry>
281     </visual>
282     <collision name="$(prefix)_bump_sensor_collision">
283       <geometry>
284         <mesh filename="package://robotiq_85_description/meshes/collision/robotiq_85_finger_tip_link.stl">
285       </geometry>
286     </collision>
287     <inertial>
288       <mass value="0.019555" />
289       <origin xyz="0 0 0.0" />
290       <inertia ixx="0.000002" ixy="0.000000" ixz="0.000000"
291         iyy="0.000000" iyz="0.000000"
292         izz="0.000000" iyz="0.000000" izz="0.000000" />
293     </inertial>
294   </link>
295
325   <!-- CONTACT SENSOR -->
326   <gazebo reference="$(prefix)gripper_finger1_finger_tip_link">
327     <sensor name="finger_tip_contact_sensor" type="contact">
328       <updateRate>10.0</updateRate>
329       <updateRate>10.0</updateRate>
330       <material>Gazebo/Wood</material>
331       <contact>
332         <collision>$(prefix)gripper_finger1_finger_tip_link_fixed_joint_lump_$(prefix)_bump_sensor_collision_collision</collision>
333       </contact>
334       <plugin name="$(prefix)gripper_finger1_finger_tip_link_plugin" filename="libgazebo_ros_bumper.so">
335         <param name="$(prefix)gripper_finger1_finger_tip_link_contact_bumperTopicName" value="" />
336       </plugin>
337     </sensor>
338   </gazebo>

```

Figure 126 Screenshot of physics parameters setup of the object (left) and the gripper(right).

```

arm_controller_force:
  type: 'position_controllers/CartesianForceController'
  end_effector_link: 'camera_link'
  robot_base_link: 'base_link'
  ft_sensor_ref_link: 'camera_link' #wrist_3_link tool0
  joints:
    - shoulder_pan_joint
    - shoulder_lift_joint
    - elbow_joint
    - wrist_1_joint
    - wrist_2_joint
    - wrist_3_joint
  # This is new:
  hand_frame_control: False # The commanded target wrench is now with respect to the robot's base link

  solver:
    error_scale: 0.01 #0.7 0.01
    iterations: 10
    forward_dynamics:
      link_mass: 1.0 # Higher values decrease oscillations in fully stretched configurations
      #Error_scale: 0.5

  pd_gains:
    trans_x: {p: 0.05} kp 0.05
    trans_y: {p: 0.05}
    trans_z: {p: 0.05}
    rot_x: {p: 1.5}
    rot_y: {p: 1.5}
    rot_z: {p: 1.5}

arm_controller_compliance:
  type: 'position_controllers/CartesianComplianceController'
  end_effector_link: 'camera_link' #shoot_link
  robot_base_link: 'base_link'
  ft_sensor_ref_link: 'wrist_3_link'
  compliance_ref_link: 'wrist_3_link' #ori:camera_link
  target_frame_topic: 'target_frame'
  joints:
    - shoulder_pan_joint
    - shoulder_lift_joint
    - elbow_joint
    - wrist_1_joint
    - wrist_2_joint
    - wrist_3_joint
  hand_frame_control: False

  stiffness: # w.r.t. compliance_ref_link
    trans_x: 1000
    trans_y: 1000
    trans_z: 1000
    rot_x: 150
    rot_y: 150
    rot_z: 150

```

Figure 127 Screenshot of controllers in the config.

```

46 <!-- Load joint controller parameters for Gazebo -->
47 <rosparam file="$(find ur5robotiq_moveit_config)/config/gazebo_controllers.yaml" />
48 <rosparam file="/home/walter/catkin_ws/src/ur5robotiq_moveit_config/config/ros_controllers.yaml" command="load" />
49
50 <!-- load the controllers -->
51 <!-- Spawn Gazebo ROS controllers -->
52 <node name="gazebo_controller_spawner" pkg="controller_manager" type="spawner" respawn="false" output="screen" args="arm_controller gripper_controller
  joint_state_controller"/>
53 <!-- Load ROS controllers -->
54 <include file="$(find ur5robotiq_moveit_config)/launch/ros_controllers.launch"/>

```

Figure 128 Settings of controllers in launch file.


```

1  #!/usr/bin/env python3
2  import rospy
3  import gym
4  import os
5  import numpy as np
6  import pandas as pd
7  import time
8  from datetime import datetime
9
10 from stable_baselines3 import PPO # DQN, DDPG, A2C, A3C
11 from stable_baselines3.common.evaluation import evaluate_policy
12 from stable_baselines3.sac.policies import MlpPolicy
13 from stable_baselines3.common.vec_env import VecVideoRecorder, DummyVecEnv
14
15 from openai_ros.task_envs.ur5e_move import move
16 from openai_ros.openai_ros_common import StartOpenAI_ROS_Environment
17
18 from gazebo_msgs.srv import GetModelState
19
20 os.chdir('/home/walter/catkin_ws/src/ur5e_rl_training/scripts')
21
22 rospy.init_node('ur5e_train_all', anonymous=True, log_level=rospy.FATAL)
23
24 environment_name = rospy.get_param('/ur5e_v0/task_and_robot_environment_name')
25 env = StartOpenAI_ROS_Environment(environment_name)
26 #env = DummyVecEnv([lambda: env]) # The algorithms require a vectorized environment to run #ZZW 20221205 THERE is a problem here.
27
28 # The noise objects for TD3
29 n_actions = env.action_space.n
30
31 algo_list = ['PPO'] # 'ACKTR', 'PP02', 'TRPO'
32
33
34
35 model = PPO("MlpPolicy", env, verbose=1, tensorboard_log="/home/walter/catkin_ws/src/ur5e_rl_training/results/tensorboard_logs/PPO/")
36
37 now = datetime.now()
38 current_time = now.strftime("%H:%M:%S")
39 print("20230502 ZZZW: START TRAINING", "@", current_time)
40
41 start = time.time()
42 model.learn(total_timesteps=200, verbose=2)
43 end = time.time()
44 training_time = ((end-start)*1000)
45 print("The training took time: ", training_time, "ms")
46 model.save("/home/walter/catkin_ws/src/ur5e_rl_training/results/trained_models/A2C")
47 print("The model has been saved.")
48 env.close()

```

Figure 129 Screenshot of RL training code.

```

1 ur5e_v0: #namespace
2
3   n_actions: 2 # Increase and decrease 7 joints.
4
5   movement_delta: 0.01 # Movement Variation of the TCP position for each step in meters.
6
7   work_space: # 3D cube in which UR5e TCP is allowed to move in
8     x_max: 1
9     x_min: 0
10    y_max: 1
11    y_min: 0
12    z_max: 1
13    z_min: 0
14
15    number_decimals_precision_obs: 2
16    acceptable_distance_to_ball: 0.2 # Distance(in metre) to the ball that we consider that it reached the ball.
17
18    done_reward: 100.0 # reward
19    closer_to_block_reward: 10.0 # reward
20
21    task_and_robot_environment_name: 'ur5e-v0'
22    ros_ws_abspath: '/home/walter/catkin_ws'

```

Figure 130 Config of RL training.